The Stata Journal (2021)
21, Number 3, pp. 626–658 DOI: 10.1177/1536867X211045538

randregret: A command for fitting random regret minimization models using Stata

Álvaro A. Gutiérrez-Vargas
Faculty of Economics and Business
KU Leuven
Leuven, Belgium
alvaro.gutierrezvargas@kuleuven.be
https://orcid.org/0000-0003-1319-5161

Michel Meulders
Faculty of Economics and Business
KU Leuven
Leuven, Belgium
michel.meulders@kuleuven.be
https://orcid.org/0000-0001-7042-6849

Martina Vandebroek
Faculty of Economics and Business
KU Leuven
Leuven, Belgium
martina.vandebroek@kuleuven.be
https://orcid.org/0000-0002-0317-1986

Abstract. In this article, we describe the `randregret` command, which implements a variety of random regret minimization (RRM) models. The command allows the user to apply the classic RRM model introduced in Chorus (2010, *European Journal of Transport and Infrastructure Research* 10: 181–196), the generalized RRM model introduced in Chorus (2014, *Transportation Research, Part B* 68: 224–238), and also the μRRM and pure RRM models, both introduced in van Cranenburgh, Guevara, and Chorus (2015, *Transportation Research, Part A* 74: 91–109). We illustrate the use of the `randregret` command by using stated choice data on route preferences. The command offers robust and cluster standard-error correction using analytical expressions of the score functions. It also offers likelihood-ratio tests that can be used to assess the relevance of a given model specification. Finally, users can obtain the predicted probabilities from each model by using the `randregretpred` command.

Keywords: st0649, randregret, randregret_pure, randregretpred, discrete choice models, semicompensatory behavior, random utility maximization, random regret minimization

1 Introduction

Chorus, Arentze, and Timmermans (2008) proposed an alternative to random utility maximization (RUM) (Manski 1977) discrete choice behavior by introducing a family of models rooted in regret theory (Loomes and Sugden 1982; Bell 1982) called random regret minimization (RRM). Intuitively, RRM claims that individuals base their choices between alternatives on the desire to avoid the situation where a nonchosen alternative ends up being more attractive than the chosen one, which would cause regret. Therefore, individuals are assumed to minimize anticipated regret when choosing among alternatives, in contrast to utility maximization.

2 Early model

The model proposed in Chorus, Arentze, and Timmermans (2008) assumes that decision makers (referred to as n) face a set of J alternatives (referred to as i or j indistinctly), each alternative being described in terms of the value of M attributes (referred to as m). Therefore, the value of attribute m of alternative i of individual n is denoted by x_{imn}. When the decision maker n is choosing between alternatives, he or she aims to minimize the anticipated random regret of a given alternative i. Consequently, the regret of alternative i on attribute m of individual n will be described by $R^{\max}_{i\leftrightarrow j,mn} = \max\{0, \beta_m \times (x_{jmn} - x_{imn})\}$. From this formulation, we can see two things. First, the regret is 0 when alternative j performs worse than i in terms of attribute m. Second, the regret grows as a linear function of the difference in attribute values in case alternative i performs worse than alternative j in terms of attribute m. Here the estimable parameter β_m gives the slope of the regret function for attribute m. Furthermore, the original version of RRM postulates that the systematic regret, $R^{\max}_{in}$, of a considered alternative i can then be written as in (1), taking the maximum regret over all alternatives:

$$R^{\max}_{in} = \max_{j\neq i}\left(\sum_{m=1}^{M} R^{\max}_{i\leftrightarrow j,mn}\right) = \max_{j\neq i}\left[\sum_{m=1}^{M} \max\{0, \beta_m \times (x_{jmn} - x_{imn})\}\right] \tag{1}$$

Finally, the anticipated random regret ($\text{RR}^{\max}_{in}$) is composed of the systematic regret $R^{\max}_{in}$ and an additive independent and identically distributed (i.i.d.) extreme-value distributed error ε_{in}, which represents the unobserved component in the regret: $\text{RR}^{\max}_{in} = R^{\max}_{in} + \varepsilon_{in}$. Assuming that the negative of ε_{in} is extreme-value type I distributed, and acknowledging that the minimization of the random regret is mathematically equivalent to maximizing the negative of the random regret, probabilities may be derived using the well-known multinomial logit formulation. Therefore, the choice probability associated with alternative i is defined in (2):

$$P^{\max}_{in} = \frac{\exp\left(-R^{\max}_{in}\right)}{\sum_{j=1}^{J}\exp\left(-R^{\max}_{jn}\right)} \qquad \text{for } i = 1, \ldots, J \tag{2}$$

3 Classical model

The major contribution made by Chorus (2010) is an elegant way to get rid of the two max operators on the attribute-level regret of the original version (Chorus, Arentze, and Timmermans 2008), which results in a nonsmooth likelihood function and triggers the need for customized optimization routines. Instead, in Chorus (2010), the new attribute-level regret is redefined by $R_{i\leftrightarrow j,mn} = \ln\left[1 + \exp\left\{\beta_m \times (x_{jmn} - x_{imn})\right\}\right]$. Therefore, the deterministic part of the regret of alternative i of individual n is now described by (3):

$$R_{in} = \sum_{j\neq i}^{J}\sum_{m=1}^{M} R_{i\leftrightarrow j,mn} = \sum_{j\neq i}^{J}\sum_{m=1}^{M} \ln\left[1 + \exp\left\{\beta_m \times (x_{jmn} - x_{imn})\right\}\right] \tag{3}$$

The two most important differences are as follows. First, the exterior max operator is replaced by a summation over all the alternatives, meaning that the choice maker's systematic regret considers not only the best nonchosen alternative as in Chorus, Arentze, and Timmermans (2008) but also the aggregate regret of all the others. Particularly, when the choice set is large, it does not seem quite reasonable to consider just one nonchosen alternative. Second, the replacement of the inner max operator has a mathematical justification because it is a continuously differentiable function that approximates the original max operator and will generate a smooth likelihood.

Following the same idea as in Chorus, Arentze, and Timmermans (2008), assuming that the random regret function (RR_{in}) also includes an additive i.i.d. extreme-value type I error term that captures the pure random noise and impact of omitted attributes in the regret: $\text{RR}_{in} = R_{in} + \varepsilon_{in}$. Finally, we obtain the same well-known and convenient closed-form logit formula for the choice probability given by (4). The last model is referred to as the classical RRM and is one of the models implemented in the command.

$$P_{in} = \frac{\exp\left(-R_{in}\right)}{\sum_{j=1}^{J}\exp\left(-R_{jn}\right)} \qquad \text{for } i = 1, \ldots, J \tag{4}$$

3.1 $\mathbf{R_{i\leftrightarrow j,mn}}$ as an approximation of $\mathbf{R^{max}_{i\leftrightarrow j,mn}}$

To illustrate how those two definitions of the regret differ from each other, a graph is presented in figure 1. The x axis represents the difference on an attribute m of two alternatives i and j for individual n, $(x_{jmn} - x_{imn})$, and the y axis represents the regret (r) that this difference generates conditional on $\beta_m = 1$.

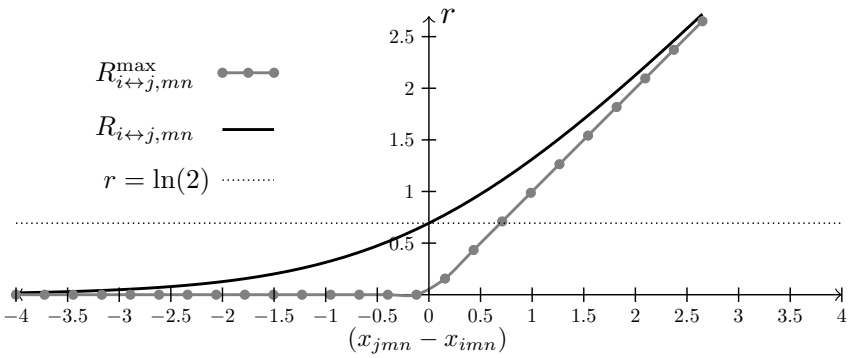

Figure 1. Comparison of $R^{\max}_{i\leftrightarrow j,mn}$ (1) and $R_{i\leftrightarrow j,mn}$ (3) conditional on $\beta_m = 1$. Adapted from Chorus (2010); reprinted with permission.

In figure 1, we can see that $R_{i\leftrightarrow j,mn}$ is a smooth version of $R^{\max}_{i\leftrightarrow j,mn}$, creating a continuous differentiable likelihood function. Also, both functions can capture semi-compensatory behavior, meaning that poor performance of a given alternative with respect to an attribute is not necessarily compensated by a good performance with respect to another attribute, which is a key feature of RRM models.

Additionally, when two alternatives have the same level for some attribute, the corresponding regret is not 0 but is equal to $\ln(2) \approx 0.69$. Though counterintuitive at first glance, note that only differences in regret or utility matter for choice probabilities (Train 2009). Hence, they remain unchanged regardless of the inclusion of this constant in the systematic regret. This can be easily checked in (4).

4 Differences between RUM and RRM models

Before we introduce three different models that generalize the underlying paradigm of the classical RRM model, we describe some essential differences between RRM and RUM while getting more insights into the RRM model.

4.1 Semicompensatory behavior and the compromise effect

Probably the most remarkable difference with the RUM model is the semicompensatory behavior that is described by RRM models. To illustrate this, we show the $R_{i\leftrightarrow j,mn}$ function with $\beta_m = 1$ in figure 2, which describes the regret generated by attribute m when a considered alternative i is being compared with alternative j, as a function of the difference between the attribute values, that is, $x_{jmn} - x_{imn}$. Segments (A) and (B) in the figure represent the magnitude of rejoice and regret, respectively, on an equal difference of attribute levels of 2.5 units. As shown in figure 2, the regret is much larger than the rejoice at an equal difference in the attribute levels. We can also see that this discrepancy becomes larger for differences with higher attribute values due to the regret function's convexity.

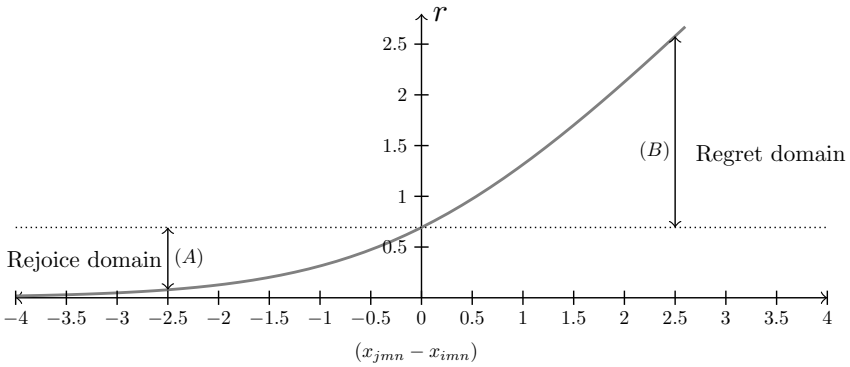

Figure 2. Semicompensatory behavior of $R_{i \leftrightarrow j,mn}$ (3) conditional on $\beta_m = 1$. Adapted from van Cranenburgh, Guevara, and Chorus (2015); reprinted with permission.

Conversely, in RUM models, linear specification of utility leads to a fully compensatory model, where the poor performance of one attribute could be compensated easily with a better performance in another attribute.

A consequence of the semicompensatory behavior of RRM models is the so-called compromise effect. Given that having an inferior performance in one attribute causes a large regret, RRM models tend to predict that alternatives with a relatively good performance in all the attributes will be preferred to alternatives with a fairly good performance in almost all attributes but a rather poor performance in just one attribute. The compromise effect has been discussed in detail by Chorus and Bierlaire (2013) and by Chorus, Rose, and Hensher (2013).

4.2 Taste parameter interpretation in RRM models

When it comes to interpretation of the RRM parameters, note that they cannot be compared with the utilitarian counterpart of RUM models. On one hand, parameters of RUM models are interpreted as the change in utility caused by an increase of a particular attribute level. On the other hand, parameters of RRM models represent the potential change in regret associated with comparing a considered alternative with another alternative in terms of the attribute, caused by one unit change in a particular attribute level. For instance, an attribute that exhibits a positive and significant coefficient suggests that regret increases as the level of that attribute increases in a nonchosen alternative compared with the level of the same attribute in the chosen one.

5 Extensions of the classical RRM model

5.1 Generalized RRM

The generalization proposed by Chorus (2014), namely, the generalized random regret minimization (GRRM) model, replaces the number 1 in the attribute-regret function (3) with a new estimable parameter $\gamma_m \in [0, 1]$, which represents the regret weight for a particular attribute. Chorus (2014) proves that depending on the value of the parameter γ_m, we can recover RUM behavior ($\gamma_m = 0$) or the classical RRM behavior ($\gamma_m = 1$), showing that GRRM is a generalization not only of the classical RRM model but also of RUM models. In our `randregret` command, detailed in section 4, we allow for only one generic and common γ for all the attributes because of its computational simplicity, which is one of the particular cases of this model implemented by Chorus (2014). Consequently, the attribute-level regret in the GRRM is described by $R^{\text{GRRM}}_{i\leftrightarrow j,mn} = \ln\left[\gamma + \exp\left\{\beta_m(x_{jmn} - x_{imn})\right\}\right]$, and the systematic part of regret in this model is given by (5):

$$R^{\text{GRRM}}_{in} = \sum_{j\neq i}^{J}\sum_{m=1}^{M} R^{\text{GRRM}}_{i\leftrightarrow j,mn} = \sum_{j\neq i}^{J}\sum_{m=1}^{M} \ln\left[\gamma + \exp\left\{\beta_m\left(x_{jmn} - x_{imn}\right)\right\}\right] \tag{5}$$

When an additive type I extreme-value i.i.d. error is added to the systematic regret function in (5), we obtain the random regret expression for the GRRM model: $\text{RR}^{\text{GRRM}}_{in} = R^{\text{GRRM}}_{in} + \varepsilon_{in}$. Finally, the choice probability of the GRRM model is presented in (6):

$$P^{\text{GRRM}}_{in} = \frac{\exp\left(-R^{\text{GRRM}}_{in}\right)}{\sum_{j=1}^{J}\exp\left(-R^{\text{GRRM}}_{jn}\right)} \qquad \text{for } i = 1, \ldots, J \tag{6}$$

An illustration of how different values of γ affect the shape of the attribute-level regret function $R^{\text{GRRM}}_{i\leftrightarrow jmn}$ is presented in figure 3.

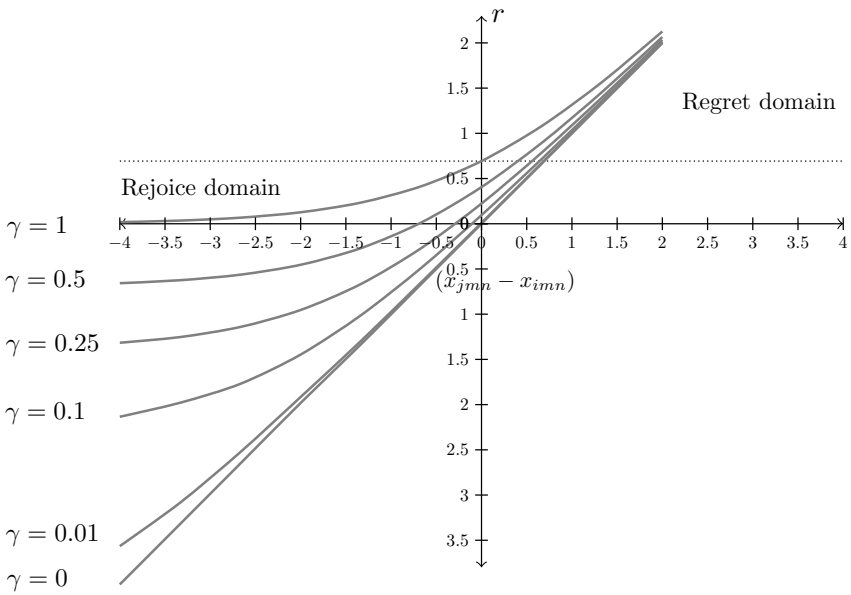

Figure 3. $R^{\text{GRRM}}_{i\leftrightarrow j,mn}$ (5) at different values of γ conditional on $\beta_m = 1$. Adapted from Chorus (2014); reprinted with permission.

As before, we have asymmetries regarding regret and rejoice produced by the difference in an attribute level. However, in the GRRM model, γ controls the convexity of $R^{\text{GRRM}}_{i\leftrightarrow j,mn}$, as can be seen in figure 3. Smaller γ values imply a less convex attribute-level regret function and consequently a smaller asymmetry between regret and rejoice. In particular, when $\gamma = 0$, the convexity of the regret function vanishes, yielding a fully compensatory behavior. Additionally, Chorus (2014) proved that the likelihood of a GRRM model with $\gamma = 0$ is equivalent to the likelihood of a linear RUM model. Finally, when $\gamma \in [0, 1]$, the sensitivity of the regret function is still higher in the regret domain but is smaller than in the classical RRM, where $\gamma = 1$.

5.2 μRRM

van Cranenburgh, Guevara, and Chorus (2015) present a new generalization of the classical RRM model that is linked to the scale parameter of the RRM model. They show that the classic regret function (3) is not scale-invariant. This property, which at first seems unfortunate, has been shown potentially useful to obtain more flexibility and also, as we will see later, for providing insights related to the observed regret in the data.

The first model proposed by van Cranenburgh, Guevara, and Chorus (2015) is the so-called μRRM model. In particular, this model is capable of estimating the scale parameter μ, which is linked to the error variance as $\text{var}(\varepsilon_i) = \left(\pi^2\mu^2/6\right)$. In this new model,

the attribute-level regret function is described by $R^{\mu\text{RRM}}_{i\leftrightarrow j,mn} = \ln[1+\exp\{(\beta_m/\mu)(x_{jmn} - x_{imn})\}]$, and consequently, the systematic regret function is given by (7):

$$\begin{aligned} R^{\mu\text{RRM}}_{in} &= \sum_{j\neq i}^{J}\sum_{m=1}^{M} \mu \times R^{\mu\text{RRM}}_{i\leftrightarrow j,mn} \\ &= \sum_{j\neq i}^{J}\sum_{m=1}^{M} \mu \times \ln\left[1+\exp\left\{(\beta_m/\mu)\left(x_{jmn} - x_{imn}\right)\right\}\right] \end{aligned} \tag{7}$$

Interestingly, in this model, we can estimate the scale parameter μ, which is well known to be nonidentifiable in the RUM context because only differences in utility matter. However, as we mentioned earlier, RRM models can describe a semicompensatory behavior, meaning that regret and rejoice do not cancel out entirely, allowing identification of the μ parameter.

As before, an additive type I extreme-value i.i.d. error term is added to the systematic regret function in (7) to obtain the random regret expression for the μRRM: $\text{RR}^{\mu\text{RRM}}_{in} = R^{\mu\text{RRM}}_{in} + \varepsilon_{in}$. Finally, the choice probabilities of this model are given by (8):

$$P^{\mu\text{RRM}}_{in} = \frac{\exp\left(-R^{\mu\text{RRM}}_{in}\right)}{\sum_{j=1}^{J}\exp\left(-R^{\mu\text{RRM}}_{jn}\right)} \qquad \text{for } i = 1,\ldots,J \tag{8}$$

van Cranenburgh, Guevara, and Chorus (2015) claim that the size of μ in the μRRM model is informative of the degree of regret imposed by the model or, stated otherwise, how much semicompensatory behavior we are observing in the decision maker's choice behavior. Given that the taste parameter β_m is divided by the scale parameter μ, then the larger the value of μ, the smaller the ratio β_m/μ and therefore the smaller the regret. Conversely, the smaller the value of μ, the bigger the ratio β_m/μ and therefore the larger the regret. This behavior is illustrated in figure 4, below, where we plotted different $R^{\mu\text{RRM}}_{i\leftrightarrow j,mn}$ for a fixed value of $\beta_m = 1$ and different values of μ.

Figure 4 shows that for arbitrarily large values of μ, the regret function becomes flatter, with the obvious consequence that the semicompensatory behavior of the model vanishes when μ tends to infinity. A formal proof of such a behavior is provided by van Cranenburgh, Guevara, and Chorus (2015), where the authors show that the μRRM model collapses into a linear RUM model when μ goes to infinity.

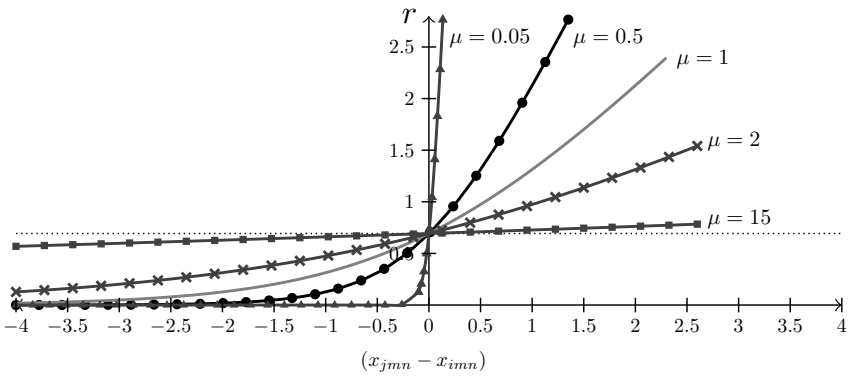

Figure 4. $R^{\mu\text{RRM}}_{i\leftrightarrow j,mn}$ (7) for different values of μ conditional on $\beta_m = 1$. Adapted from van Cranenburgh, Guevara, and Chorus (2015); reprinted with permission.

On the other hand, when the value of μ is arbitrarily small, the μRRM model represents the strongest semicompensatory behavior possible among the RRM family models. This scheme is explained in the following section.

Finally, in this model, note that the convexity of the attribute-level regret function ($R^{\mu\text{RRM}}_{i\leftrightarrow j,mn}$) is based on the taste parameter β and on μ; therefore, large values of the taste parameter could compensate large values of μ, generating a ratio close to 1.

5.3 Pure RRM

The second model proposed by van Cranenburgh, Guevara, and Chorus (2015) is a particular case of the μRRM model that is generated by arbitrarily small values of μ in the μRRM model. As explained earlier, small values of μ mean that the ratio β_m/μ is very large, causing the regret function to yield very strong differences between regrets and rejoices. This can be seen graphically in figure 4, where the smaller the value of μ, the larger the slope of the regret function within the regret domain.

Interestingly, the authors formally proved that for μ going to 0 in the μRRM model in (7), the model collapses into a linear specification (van Cranenburgh, Guevara, and Chorus 2015, appendix D). The authors call the resulting model the pure-RRM (hereafter PRRM) model, which describes the strongest semicompensatory behavior of all RRM models. The specification of systematic regret imposed by the PRRM model is presented in (9) and (10):

$$R^{\text{PRRM}}_{in} = \sum_{m=1}^{M} \beta_m x^{\text{PRRM}}_{imn} \tag{9}$$

$$x^{\text{PRRM}}_{imn} = \begin{cases} \sum_{j\neq i}^{J} \max\left(0, x_{jmn} - x_{imn}\right) & \text{if } \beta_m > 0 \\ \sum_{j\neq i}^{J} \min\left(0, x_{jmn} - x_{imn}\right) & \text{if } \beta_m < 0 \end{cases} \tag{10}$$

From (10), we can see that the PRRM model can be understood as a traditional logit model using transformed attribute levels. Notice here that to estimate the PRRM, we need to know the sign of the attributes a priori. In some situations, this requisite is not very restrictive. For instance, in transport contexts where the alternatives are mainly described in terms of its travel time (tt) and total cost (tc), we can expect the coefficients β_{tc} and β_{tt} to have negative signs, given that cheaper and faster routes are preferred to costlier and slower ones. The negative sign can be understood in terms of regret as follows: when the total time (total cost) in nonchosen alternatives increases, our regret decreases, given that the chosen alternative becomes relatively faster (cheaper).

Finally, adding the usual additive i.i.d. type I extreme-value error as in the previous models to the systematic regret in (9), we obtain the random regret of the model: $\text{RR}_{in}^{\text{PRRM}} = R_{in}^{\text{PRRM}} + \varepsilon_{in}$. Consequently, the choice probability of the PRRM model under the stated distributional assumption is given by (11):

$$P_{in}^{\text{PRRM}} = \frac{\exp\left(-R_{in}^{\text{PRRM}}\right)}{\sum_{j=1}^{J} \exp\left(-R_{jn}^{\text{PRRM}}\right)} \qquad \text{for } i = 1, \ldots, J \tag{11}$$

6 Alternative-specific constants

The inclusion of alternative-specific constants (ASC) in the presented models is possible by simply adding them into the systematic part of the regret. To exemplify this, let R_{in}^{*} denote a generic systematic regret of alternative i as defined in (3), (5), (7), or (9). We denote by α_i the ASC of alternative i in (12). The inclusion of the ASC serves the same purpose as in RUM models, which is to account for omitted attributes for a particular alternative. As usual, for identification purposes, we need to exclude one of the ASC from the model specification. For a detailed discussion of ASC in the context of RRM models, see van Cranenburgh and Prato (2016).

$$R_{in}^{*} = \sum_{j \neq i}^{J} \sum_{m=1}^{M} R_{i \leftrightarrow j, mn}^{*} + \alpha_i \tag{12}$$

7 Relationships among the different models

In figure 5, we present the relationships among all the presented models. Solid arrows state that a model collapses onto another model for a specific value of some parameter. For instance, we can see the connection between the classical RRM model and the GRRM model when $\gamma = 1$. On the other hand, dotted arrows indicate that the choice probabilities and the likelihood of two models are the same, but not necessarily the estimated parameters. For instance, Chorus (2014) showed that the relationship among RUM and RRM parameters is described by $\beta_m^{\text{RRM}} = J \times \beta_m^{\text{RUM}}$ when $\gamma = 0$, where J is the size of the choice set. Similarly, van Cranenburgh, Guevara, and Chorus (2015) showed that when μ goes to infinity the relationship is described by $\beta_m^{\text{RUM}} \cong (J/2) \times \beta_m^{\text{RRM}}$.

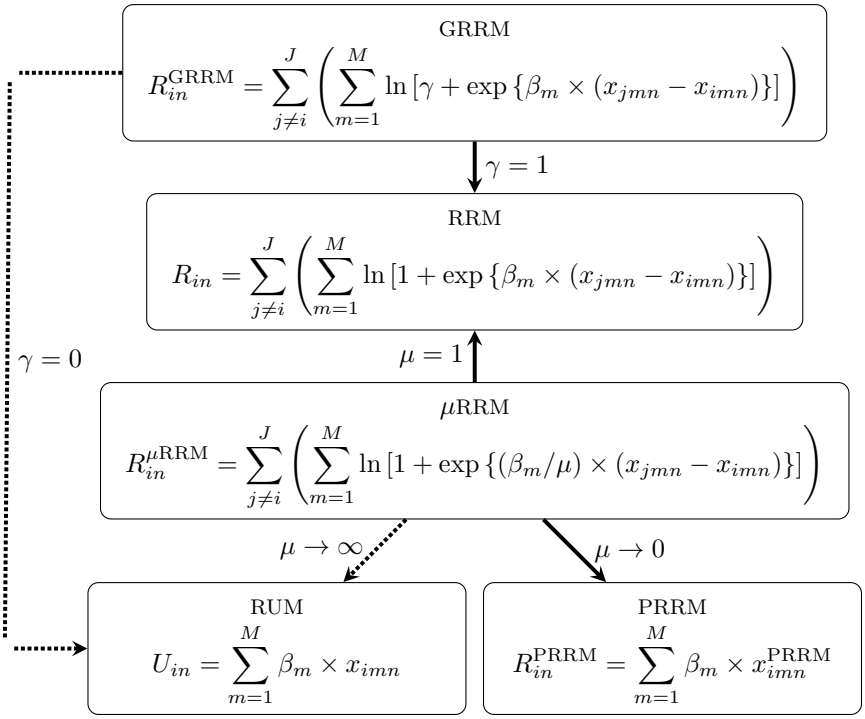

Figure 5. Interrelationship among the models based on parameters

The relationships in figure 5 allow us to use a likelihood-ratio (LR) test to compare nested models and check which model fits the data best. In particular, table 1 lists the relevant hypotheses with the corresponding LR statistic and the asymptotic distribution of the test. The first column lists the models that we can compare based on a particular parameter. The second column lists the formal hypotheses for the relevant parameter. The third column presents the LR statistic in each case, where $\ell(\cdot)$ represents the log likelihood of the model and $\widehat{\boldsymbol{\theta}}_{\text{RRM}}$, $\widehat{\boldsymbol{\theta}}_{\text{GRRM}}$, $\widehat{\boldsymbol{\theta}}_{\mu\text{RRM}}$, and $\widehat{\boldsymbol{\theta}}_{\text{RUM}}$ represent the full set of parameters of the classical RRM, GRRM, μRRM, and linear-in-parameters RUM models, respectively. Finally, the fourth column lists the asymptotic distribution of the statistic under the null hypothesis. The fact that the two first hypotheses follow a different distribution from the traditional χ^2_1 is because we are testing a null hypothesis on the boundary of the parametric space of γ. For details about deriving the distribution of the LR test under nonstandard conditions, see Self and Liang (1987). Additionally, illustrations of this matter can be found in Molenberghs and Verbeke (2007) and Gutierrez, Carter, and Drukker (2001).

Table 1. LR test for model comparison

Models	Hypothesis	LR statistic	Distribution under H_0
RRM versus GRRM	$H_0: \gamma = 1$ $H_1: \gamma < 1$	$2\left\{\ell\left(\widehat{\boldsymbol{\theta}}_{\text{GRRM}}\right) - \ell\left(\widehat{\boldsymbol{\theta}}_{\text{RRM}}\right)\right\}$	$0.5\left(\chi^2_0 + \chi^2_1\right)$
RUM versus GRRM	$H_0: \gamma = 0$ $H_1: \gamma > 0$	$2\left\{\ell\left(\widehat{\boldsymbol{\theta}}_{\text{GRRM}}\right) - \ell\left(\widehat{\boldsymbol{\theta}}_{\text{RUM}}\right)\right\}$	$0.5\left(\chi^2_0 + \chi^2_1\right)$
RRM versus μRRM	$H_0: \mu = 1$ $H_1: \mu \neq 1$	$2\left\{\ell\left(\widehat{\boldsymbol{\theta}}_{\mu\text{RRM}}\right) - \ell\left(\widehat{\boldsymbol{\theta}}_{\text{RRM}}\right)\right\}$	χ^2_1

The `randregret` command always fits the classical RRM model to use those estimates as starting points for the extended versions of the model, GRRM and μRRM. The LR tests for $\gamma = 1$ and $\mu = 1$ do not require extra computations. However, for testing $\gamma = 0$, an additional linear RUM model is fit. Regardless, the user has the option to deactivate the tests to speed up computations if desired.

Additionally, it is worth mentioning that the presented asymptotic distributions of the LR test are only valid when no robust or cluster corrected variance–covariance matrices are applied. If said corrections are used, a Wald test should be applied instead. This test can be straightforwardly implemented using the postestimation command `test`.

8 Robust standard errors

The use of robust standard errors corrected by cluster in discrete choice models is a common practice given the panel structure that is created when an individual answers multiple-choice situations in state-preference surveys. To illustrate this, we can write our maximum-likelihood estimation equations as in (13). Where $\boldsymbol{\theta}$ is the full set of parameters, $\boldsymbol{S}(\boldsymbol{\theta}; y_n, \boldsymbol{x}_n) = \partial \ln L_n / \partial \boldsymbol{\theta}$ represents the score functions, $\ln L_n$ is the log likelihood of observation n, $\boldsymbol{x}_n$ is the full set of attributes, and y_n is the response variable that takes the value of 1 when alternative i is selected and 0 otherwise.

$$G(\boldsymbol{\theta}) = \sum_{n=1}^{N} \boldsymbol{S}(\boldsymbol{\theta}; y_n, \boldsymbol{x}_n) = \mathbf{0} \tag{13}$$

We can compute the robust variance estimator of $\boldsymbol{\theta}$ using (14), where $\boldsymbol{D} = -\mathbf{H}^{-1}$ is the negative of the inverse of the Hessian resulting from the optimization procedure, and $\boldsymbol{u}_n = \boldsymbol{S}(\widehat{\boldsymbol{\theta}}; y_n, \boldsymbol{x}_n)$ are row vectors that contain the score functions evaluated at $\widehat{\boldsymbol{\theta}}$.

$$\widehat{V}\left(\widehat{\boldsymbol{\theta}}\right) = \boldsymbol{D}\left(\frac{n}{n-1}\sum_{n=1}^{N} \boldsymbol{u}_n' \boldsymbol{u}_n\right)\boldsymbol{D} \tag{14}$$

Equation (14) is appropriate only if the observations are independent. However, when the same individual answers several choice situations, we can expect some degree of dependency. When such a structure is present in the data, a more appropriate robust variance estimate is given by (15), where C_k contains the indices of all observations belonging to the same individual k for $k = 1, 2, \ldots, n_c$, with n_c being the total number of different individuals present in the dataset.

$$\widehat{V}\left(\widehat{\boldsymbol{\theta}}\right) = \boldsymbol{D}\left\{\frac{n_c}{n_c - 1}\sum_{k=1}^{n_c}\left(\sum_{n\in C_k}\boldsymbol{u}_n\right)'\left(\sum_{n\in C_k}\boldsymbol{u}_n\right)\right\}\boldsymbol{D} \tag{15}$$

Appendix A provides details on the analytical form of the scores by each model presented in this article. Additionally, the `randregret` command can compute corrected standard errors by using the analytical expressions of the score functions without relying on numerical approximations.

9 Commands

9.1 randregret

9.1.1 Syntax

`randregret` *depvar* [*indepvars*] [*if*] [*in*], `rrmfn(`*string*`)` `group(`*varname*`)`
`alternatives(`*varname*`)` [`positive(`*varlist*`)` `negative(`*varlist*`)`
`basealternative(#)` `noconstant` `uppermu(#)` `showancillary` `notlr`
`initgamma(#)` `initmu(#)` `cluster(`*varname*`)` `robust` `level(#)`
maximize_options]

9.1.2 Description

`randregret` is implemented as a Mata-based `d0 ml` evaluator. The command allows one to implement four different regret functions.

9.1.3 Options

`rrmfn(`*string*`)` is required and specifies the regret function that will be used. `classic` uses the systematic regret of (3), `gene` uses (5), `mu` uses (7), and `pure` uses (9). The last option will use the `randregret_pure` command (see section 9.2) to create the transformed alternative-specific attributes.

`group(`*varname*`)` is required and specifies a numeric identifier variable for the choice situation.

`alternatives(`*varname*`)` is required and specifies a numeric identifier variable of the alternative for each choice situation.

`positive(`*varlist*`)` specifies to include attributes with an assumed positive sign when performing `rrmfn(pure)`. Either `positive()` or `negative()` is required when using `rrmfn(pure)`.

`negative(`*varlist*`)` specifies to include attributes with an assumed negative sign when performing `rrmfn(pure)`. Either `negative()` or `positive()` is required when using `rrmfn(pure)`.

`basealternative(#)` sets the reference level in dummy coding for ASC.

`noconstant` suppresses the ASC from the specification.

`uppermu(#)` alters the optimization procedure using an ancillary parameter on the logit scale searching for μ in the space $[0, \#]$. The default is `uppermu(5)`.

`showancillary` specifies to show the value of the estimated ancillary parameter when performing `rrmfn(gene)` or `rrmfn(mu)`.

`notlr` specifies to suppress the computations of the LR test over γ and μ, respectively, when performing `rrmfn(gene)` or `rrmfn(mu)`.

`initgamma(#)` specifies to set the initial value for the ancillary parameter for γ when performing `rrmfn(gene)`. The default is `initgamma(0)`.

`initmu(#)` specifies to set the initial value for the ancillary parameter for μ when performing `rrmfn(mu)`. The default is `initmu(0)`.

`cluster(`*varname*`)` specifies to adjust the variance–covariance matrix using (15) computing clusters across individuals answering multiple questions.

`robust` specifies to adjust the variance–covariance matrix using (14).

`level(#)` sets the confidence level. The default is `level(95)`.

maximize_options: `difficult`, `iterate(#)`, `trace`, `gradient`, `showstep`, `hessian`, `tolerance(#)`, `ltolerance(#)`, `gtolerance(#)`, `nrtolerance(#)`, `technique(`*algorithm_spec*`)`, `from(`*init_specs*`)`; see [R] **Maximize**. `technique(bhhh)` is not allowed.

9.2 randregret_pure

9.2.1 Syntax

`randregret_pure` *varlist* [*if*] [*in*], `group(`*varname*`)` `signbeta(`*string*`)` `prefix(`*stubname*`)`

9.2.2 Description

`randregret_pure` implements the alternative-specific attribute transformations required to fit the PRRM model in (10).

9.2.3 Options

`group(`*varname*`)` is required and specifies a numeric identifier variable for the choice situation.

`signbeta(`*string*`)` is required and specifies the sign of all the alternative-specific attributes included in *varlist*. Specifying `pos` indicates that the sign of the attributes is positive, and specifying `neg` indicates that the sign is negative.

`prefix(`*stubname*`)` is required and specifies the prefix of the new transformed alternative-specific attributes that `randregret_pure` will create.

9.3 randregretpred

9.3.1 Syntax

`randregretpred` *newvar* [*if*] [*in*], `group(`*varname*`)` `alternatives(`*varname*`)` [`proba xb`]

9.3.2 Description

`randregretpred` can be invoked after `randregret` to obtain both predicted probabilities and systematic estimated regret. `randregretpred` automatically identifies the last fitted model, calculates the predicted-choice probabilities recovering the parameters obtained from the likelihood maximization, and then plugs them back in using (4), (6), (8), or (11) depending on the previously fitted model. Additionally, it is also possible to recover the linear prediction of the systematic regret from (3), (5), (7), or (9).

9.3.3 Options

`group(`*varname*`)` is required and specifies a numeric identifier variable for the choice situation.

`alternatives(`*varname*`)` is required and specifies a numeric identifier variable of the alternative for each choice situation.

`proba`, the default, calculates the predicted-choice probabilities of each alternative.

`xb` calculates the linear prediction in the regret function.

10 Examples

To show the use of the `randregret` command, we use data from van Cranenburgh (2018) that correspond to a value of time–stated choice experiment. The choice situation in this experiment consisted of three unlabeled route alternatives, each consisting of two

generic attributes: travel cost (`tc`) and travel time (`tt`). In this experiment, each respondent answered a total of 10 choice situations. Table 2 presents the first choice situation presented to respondents in the stated choice experiment. The authors used a so-called D-efficient design to optimize the statistical efficiency of the experiment.[1]

Table 2. English translation of first choice situation

Attribute	Route A	Route B	Route C
Travel time (one way)	23 min.	27 min.	35 min.
Travel cost (one way)	6 euros	4 euros	3 euros

The following variables will be used in our specifications of `randregret`:

- `altern`: ID of the alternative faced by the user.
- `choice`: whether the alternative was chosen by an individual (0–1 dummy).
- `id`: ID of the individual.
- `cs`: ID of the choice situation faced by the individual.
- `tt`: Total travel time (one way) of alternative i in minutes.
- `tc`: Total travel cost (one way) of alternative i in euros.

The data setup for `randregret` is equivalent to that used by `clogit` (see [R] **clogit**) and the latest released command `cmclogit` (see [CM] **cmclogit**), meaning it has a panel representation in terms of individual–alternative, that is to say, in long format. The data are loaded from the server to Stata directly by using `import delimited` and the URL given in van Cranenburgh (2018). The data are currently in wide format, and just for the sake of illustration, we show the data manipulations required to use `randregret`. We list the first four choice situations answered by the first individual with the corresponding alternative-specific attributes, total time and total cost.

1. The complete experimental design can be found in appendix A of van Cranenburgh and Alwosheel (2019).

```
. scalar server = "https://data.4tu.nl/ndownloader/"
. scalar doi = "files/24015353"
. import delimited "`=server + doi'", clear
(encoding automatically selected: ISO-8859-1)
(29 vars, 1,060 obs)
. keep obs id cs tt1 tc1 tt2 tc2 tt3 tc3 choice
. list obs id cs tt1 tc1 tt2 tc2 tt3 tc3 choice in 1/4, sepby(obs)
```

	obs	id	cs	tt1	tc1	tt2	tc2	tt3	tc3	choice
1.	1	1	1	23	6	27	4	35	3	3
2.	2	1	2	27	5	35	4	23	6	2
3.	3	1	3	35	3	23	5	31	4	1
4.	4	1	4	27	4	23	5	35	3	3

Given that `randregret` requires the data to be presented in long format, we will perform the required transformation by using the `reshape` command and present the same information in long format.

```
. rename (choice) (choice_w)
. reshape long tt tc, i(obs) j(altern)
(j = 1 2 3)
Data                                Wide   ->   Long
-----------------------------------------------------------------------------
Number of observations             1,060   ->   3,180
Number of variables                   10   ->   7
j variable (3 values)                      ->   altern
xij variables:
                              tt1 tt2 tt3   ->   tt
                              tc1 tc2 tc3   ->   tc
-----------------------------------------------------------------------------
. generate choice = 0
. replace choice = 1 if choice_w==altern
(1,060 real changes made)
. label define alt_label 1 "First" 2 "Second" 3 "Third"
. label values altern alt_label
```

```
. list obs altern choice id cs tt tc in 1/12, sepby(obs)
```

	obs	altern	choice	id	cs	tt	tc
1.	1	First	0	1	1	23	6
2.	1	Second	0	1	1	27	4
3.	1	Third	1	1	1	35	3
4.	2	First	0	1	2	27	5
5.	2	Second	1	1	2	35	4
6.	2	Third	0	1	2	23	6
7.	3	First	1	1	3	35	3
8.	3	Second	0	1	3	23	5
9.	3	Third	0	1	3	31	4
10.	4	First	0	1	4	27	4
11.	4	Second	0	1	4	23	5
12.	4	Third	1	1	4	35	3

After the data manipulation, we can fit the four different RRM models that the command `randregret` can estimate. Before going into the details of each possible specification of the regret function, we will discuss two required options for every model: `group()` and `alternatives()`. The `group()` option contains an identifier of each choice situation in the sample, which in our case corresponds with the variable `obs`. The `alternatives()` option identifies the available alternatives of the choice set, which in our case corresponds with the variable `altern`.

We start with the classical RRM that uses (3) as systematic regret. To obtain such a model, we need to specify `rrmfn(classic)`. Additionally, we declare `noconstant` because alternatives were nonlabeled in the survey; therefore, we suppress the ASC. Here we can see that, as expected, both variables' coefficients are negative and highly significant. This latter result can be interpreted as follows. A negative and significant coefficient suggests that regret decreases as the level of that attribute increases in a nonchosen alternative compared with the same attribute level in a chosen one. For example, a negative coefficient estimated for the attribute "total time" indicates that the regret decreases as the total time increases in a nonchosen alternative, compared with the level of the chosen option. The same interpretation can be made for the attribute "total cost".

```
. randregret choice tc tt, group(obs) alternatives(altern) rrmfn(classic)
> noconstant
```

```
Fitting Classic RRM Model

initial:       log likelihood =  -1164.529
alternative:   log likelihood = -1156.5784
rescale:       log likelihood =   -1121.29
Iteration 0:   log likelihood =   -1121.29
Iteration 1:   log likelihood = -1118.4843
Iteration 2:   log likelihood = -1118.4784
Iteration 3:   log likelihood = -1118.4784

RRM: Classic Random Regret Minimization Model
Case ID variable: obs                         Number of cases    =       1060
Alternative variable: altern                  Number of obs      =       3180
                                              Wald chi2(2)       =     114.72
Log likelihood = -1118.4784                   Prob > chi2        =     0.0000
```

choice	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
RRM						
tc	-.417101	.0399883	-10.43	0.000	-.4954767	-.3387253
tt	-.102813	.0099862	-10.30	0.000	-.1223857	-.0832403

However, given that we observe multiple answers from each individual in the presented data, we need to correct our standard errors considering this panel structure. We can easily cluster our standard errors across individuals by using the `cluster(id)` option, which implements the cluster–robust variance–covariance matrix described in (15). When we refit the model using the robust cluster correction, we can see a considerable increase in the standard error. This change can mainly be explained because the cluster correction treats every set of 10 answers from each of the 106 individuals as dependent observations, which is different from the latter, which assumed each of the 1,060 choice situations were all independent. We will present the following models by using robust standard errors with clusters across individuals.

```
. randregret choice tc tt, group(obs) alternatives(altern) rrmfn(classic)
> noconstant cluster(id) nolog
```

```
Fitting Classic RRM Model

RRM: Classic Random Regret Minimization Model
Case ID variable: obs                         Number of cases    =       1060
Alternative variable: altern                  Number of obs      =       3180
                                              Wald chi2(2)       =      40.41
Log likelihood = -1118.4784                   Prob > chi2        =     0.0000
                                 (Std. Err. adjusted for   106 clusters in id)
```

choice	Coefficient	Robust std. err.	z	P>|z|	[95% conf.	interval]
RRM						
tc	-.417101	.068059	-6.13	0.000	-.5504943	-.2837078
tt	-.102813	.0182526	-5.63	0.000	-.1385874	-.0670386

To fit the GRRM model, we simply need to declare `rrmfn(gene)` to use (5) as our systematic regret function. From the output, we can observe that `randregret` fit three models. The classical RRM (3) is fit to use the parameters as starting points for the GRRM, and it is used in the LR test of $\gamma = 1$. Afterward, a linear RUM model was fit to obtain the constrained likelihood for the LR test of $\gamma = 0$. Finally, the GRRM model was fit.

Because γ must lie between 0 and 1, the optimization uses an ancillary parameter with a logistic transformation during the optimization procedure: $\gamma = \exp(\gamma^*)/\{1 + \exp(\gamma^*)\} = \text{logit}^{-1}(\gamma^*) = \text{invlogit}(\gamma^*)$, where γ^* is an unbounded ancillary parameter. Normally, γ^* is hidden from the output, but it can be shown using the option `showancillary`. The resulting γ^* is displayed in the `_cons` variable of the `gamma_star` equation.

```
. randregret choice tc tt, group(obs) alternatives(altern) rrmfn(gene)
> noconstant cluster(id) showancillary
------------------------------------------------------------------------------

Fitting Classic RRM for Initial Values
------------------------------------------------------------------------------

initial:       log likelihood =  -1164.529
alternative:   log likelihood = -1156.5784
rescale:       log likelihood =   -1121.29
Iteration 0:   log likelihood =   -1121.29
Iteration 1:   log likelihood = -1118.4843
Iteration 2:   log likelihood = -1118.4784
Iteration 3:   log likelihood = -1118.4784
------------------------------------------------------------------------------

Fitting Conditional Logit as a Restricted Model (gamma=0) for LR test
------------------------------------------------------------------------------

------------------------------------------------------------------------------

Fitting Generalized RRM Model
------------------------------------------------------------------------------

initial:       log likelihood = -1120.7001
rescale:       log likelihood = -1120.7001
rescale eq:    log likelihood = -1120.7001
Iteration 0:   log likelihood = -1120.7001
Iteration 1:   log likelihood = -1118.5366
Iteration 2:   log likelihood = -1118.3484
Iteration 3:   log likelihood = -1118.3307
Iteration 4:   log likelihood = -1118.3302
Iteration 5:   log likelihood = -1118.3302
```

```
GRRM: Generalized Random Regret Minimization Model
Case ID variable: obs                         Number of cases    =       1060
Alternative variable: altern                  Number of obs      =       3180
                                              Wald chi2(2)       =      10.23
Log likelihood = -1118.3302                   Prob > chi2        =     0.0060
                                 (Std. Err. adjusted for   106 clusters in id)
------------------------------------------------------------------------------
      choice | Coefficient  Robust std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
RRM          |
          tc |  -.3904872   .1248997    -3.13   0.002    -.6352861   -.1456884
          tt |  -.0967528   .0307009    -3.15   0.002    -.1569255     -.03658
-------------+----------------------------------------------------------------
gamma_star   |
       _cons |   1.291135   3.303988     0.39   0.696    -5.184563    7.766832
-------------+----------------------------------------------------------------
       gamma |   .7843392   .5588736                      .0055712    .9995766
------------------------------------------------------------------------------
LR test of gamma=0: chibar2(01) = 9.41                Prob >= chibar2 = 0.001
LR test of gamma=1: chibar2(01) = 0.30                Prob >= chibar2 = 0.293
------------------------------------------------------------------------------
```

Finally, using $\widehat{\gamma^*}$, we can obtain $\widehat{\gamma}$ back on the original scale from 0 to 1 by using the logistic transformation. It is displayed as `gamma` in the output. The standard error of `gamma` is computed using the delta method. To exemplify the last point, manually, it is possible to recover $\widehat{\gamma}$ (`gamma`) using (see [R] **nlcom**).

```
. nlcom (gamma: invlogit(_b[gamma_star:_cons]))
       gamma: invlogit(_b[gamma_star:_cons])
------------------------------------------------------------------------------
      choice | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
       gamma |   .7843392   .5588736     1.40   0.160    -.3110331    1.879711
------------------------------------------------------------------------------
```

From the `nlcom` output, some immediate discrepancies are evident regarding the confidence interval (CI), where the upper bound violates the restriction that we imposed on γ. As Buis (2014) documented using the `heckman` command, the discrepancies in CI occur because `nl` computes the CI using $\widehat{\gamma} \pm z_{\rho}\{\widehat{\text{Var}}(\widehat{\gamma})\}^{1/2}$. Accordingly, using the procedure of Buis (2007) and noting that the $\widehat{\gamma}$ and $\{\widehat{\text{Var}}(\widehat{\gamma})\}^{1/2}$ are stored in `e(gamma)` and `e(gamma_sd)`, respectively, we can recover the CI of `nlcom` as follows.

```
. display "confidence interval for gamma from nl: [" /*
> */ e(gamma) - e(gamma_se)*invnormal(.975) ", "     /*
> */ e(gamma) + e(gamma_se)*invnormal(.975) "]"
confidence interval for gamma from nl: [-.31103306, 1.8797114]
```

On the other hand, `randregret` computes the CI using the endpoints of $\widehat{\gamma^*}$ and then transforms those endpoints back into the restricted space. Therefore, the CI is computed using $\text{invlogit}[\widehat{\gamma^*} \pm z_\rho\{\widehat{\text{Var}}(\widehat{\gamma}^*)\}^{1/2}]$. We can replicate the CI produced by `randregret` manually as follows:

```
. display "confidence interval for gamma from randregret: ["   /*
> */ invlogit(_b[gamma_star:_cons] -                           /*
> */ invnormal(.975)* _se[gamma_star:_cons]) ", "              /*
> */ invlogit(_b[gamma_star:_cons] +                           /*
> */ invnormal(.975)* _se[gamma_star:_cons]) "]"
confidence interval for gamma from randregret: [.00557117, .99957663]
```

Even though both ways are asymptotically equivalent, in finite samples they are likely to differ. Moreover, the way used by `randregret` ensures that the restrictions imposed on the parameter are met by the CI too.

Additionally, `randregret` computes two LR tests for γ. As explained earlier, given that we are testing a null hypothesis at the boundaries of the parametric space, we need to adjust the critical value (Gutierrez, Carter, and Drukker 2001). This is why the test mentions a `chibar2(01)`, which is a mixture of a χ^2_1 (50%) and a χ^2_0 (50%). From the test, we can see that the hypothesis for $\gamma = 0$ is rejected, meaning that there is statistical evidence in favor of the data being generated by regret minimization behavior and not from RUM. Besides, we can see that the hypothesis for $\gamma = 1$ cannot be rejected, meaning here that the GRRM model is not significantly different from the classical RRM. Finally, it is important to note that, as stated before, the conclusions derived from the LR test that `randregret` displays by default are only valid when no corrections to the variance–covariance matrix are implemented, which does not happen in this case. Hence, a Wald test is better suited in this context.

The μRRM model can be obtained by typing `rrmfn(mu)`, implementing (7) as systematic regret. For this model, similar to the GRRM model, we use an ancillary parameter approach to bound the searching space of our algorithm for μ between 0 and M. The transformation used is $\mu = M \times [\exp(\mu^*)/\{1 + \exp(\mu^*)\}] = M \times \{\text{logit}^{-1}(\mu^*)\} = M \times \{\text{invlogit}(\mu^*)\}$, where μ^* is an unbounded ancillary parameter and M is equal to the upper bound of the searching space. The upper bound that we used in this case was equal to 10, as can be seen in the `uppermu()` option. From the output, we see that `randregret` first runs the classical RRM model and uses the common parameters with the μRRM model as starting points for the maximization procedure.

```
. local up = 10
. randregret choice tc tt, group(obs) alternatives(altern) rrmfn(mu) noconstant
> uppermu(`up') showancillary cluster(id)
------------------------------------------------------------------------------
Fitting Classic RRM for Initial Values
------------------------------------------------------------------------------

initial:       log likelihood =  -1164.529
alternative:   log likelihood = -1156.5784
rescale:       log likelihood =   -1121.29
Iteration 0:   log likelihood =   -1121.29
Iteration 1:   log likelihood = -1118.4843
Iteration 2:   log likelihood = -1118.4784
Iteration 3:   log likelihood = -1118.4784
------------------------------------------------------------------------------
Fitting muRRM Model
------------------------------------------------------------------------------

initial:       log likelihood = -1121.2577
rescale:       log likelihood = -1121.2577
rescale eq:    log likelihood = -1121.2577
Iteration 0:   log likelihood = -1121.2577  (not concave)
Iteration 1:   log likelihood = -1118.9528
Iteration 2:   log likelihood = -1118.5884
Iteration 3:   log likelihood =  -1118.398
Iteration 4:   log likelihood = -1118.3965
Iteration 5:   log likelihood = -1118.3965
muRRM: Mu-Random Regret Minimization Model
Case ID variable: obs                         Number of cases    =       1060
Alternative variable: altern                  Number of obs      =       3180
                                              Wald chi2(2)       =      66.95
Log likelihood = -1118.3965                   Prob > chi2        =     0.0000
                                 (Std. Err. adjusted for   106 clusters in id)
------------------------------------------------------------------------------
             |               Robust
      choice | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
RRM          |
          tc |  -.4280409   .0557747    -7.67   0.000    -.5373572   -.3187245
          tt |  -.1059436   .0152902    -6.93   0.000    -.1359119   -.0759754
-------------+----------------------------------------------------------------
mu_star      |
       _cons |    -2.0056   .7911288    -2.54   0.011    -3.556183   -.4550157
-------------+----------------------------------------------------------------
          mu |   1.186163   .8270969                      .2775523    3.881689
------------------------------------------------------------------------------
LR test of mu=1: chi2(1) =0.16                        Prob >= chibar2 = 0.686
------------------------------------------------------------------------------
```

The resulting $\widehat{\mu}^*$ is displayed in the `_cons` variable of the `mu_star` equation because of the `showancillary` option. To recover the value of $\widehat{\mu}$, `randregret` applies the transformation described above. The same procedure can be performed using `nlcom` in the same fashion as we did for the GRRM model, with the only difference being that we need to multiply by the defined upper bound of the searching space to recover the parameter $\widehat{\mu}$ correctly. The same discrepancies in the CI produced by `nlcom` are due to the matter explained earlier in the GRRM model context and can be addressed as stated.

```
. nlcom (mu :invlogit(_b[mu_star:_cons])*`up')
         mu: invlogit(_b[mu_star:_cons])*10
```

choice	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
mu	1.186163	.8270969	1.43	0.152	-.4349177	2.807243

Additionally, `randregret` shows the LR test results for testing $\mu = 1$. We see that it is not possible to reject the hypothesis that μ is equal to 1 ($p = 0.686$), meaning that the model is statistically equivalent to the classical RRM. However, because we fit the model using a cluster–robust variance–covariance matrix, the inference derived from LR tests is no longer valid, and a Wald test should be performed instead. An important remark for practitioners is that if the maximum is reached at $\widehat{\mu} = M$, it is highly likely that μ is tending to infinity. Therefore, as argued in van Cranenburgh, Guevara, and Chorus (2015), this fact suggests that there is evidence in favor that the choice behavior is better represented by a linear RUM model.

Finally, the PRRM model can be fit using the `rrmfn(pure)` option. As we mentioned before, this model is a particular case of the μRRM with μ arbitrarily small, and it is described by (9) and (10). Important differences with the common syntax need to be mentioned. Given that we need to feed the model with the expected signs of attributes, we do not include the explanatory variables conventionally. Instead, we split the attributes between the ones with an assumed positive sign and the ones with an assumed negative sign in the options `positive()` and `negative()`, respectively. In this particular case, both of our attributes are expected to have a negative sign because faster and cheaper routes are preferable to slower and costlier ones. Therefore, when the level on a nonchosen alternative increases, the regret decreases. Consequently, we need to include the two attributes as follows: `negative(tc tt)`.

```
. randregret choice, negative(tc tt) group(obs) alternatives(altern) rrmfn(pure)
> noconstant cluster(id)
PRRM: Pure Random Regret Minimization Model
Case ID variable: obs                          Number of cases    =       1060
Alternative variable: altern                   Number of obs      =       3180
                                               Wald chi2(2)       =      21.06
Log likelihood = -1128.3777                    Prob > chi2        =     0.0000
                                  (Std. err. adjusted for 106 clusters in id)
```

choice	Coefficient	Robust std. err.	z	P>|z|	[95% conf.	interval]
choice						
tc	-.285628	.0647545	-4.41	0.000	-.4125446	-.1587114
tt	-.0661575	.0169355	-3.91	0.000	-.0993505	-.0329645

```
The Pure-RRM uses a transformation of the original regressors using options
positive() and negative() as detailed in S. van Cranenburgh et. al (2015)
Afterward, randregret invokes clogit using these transormed regresors
```

As mentioned in the footnote of the output, `randregret` performs the attribute transformation using (10). The transformed attributes are generated using the command `randregret_pure`. Afterwards, `randregret` simply invokes `clogit` to fit the model using these transformed attributes.

Additionally, for the sake of illustration, we will also fit the PRRM model using `randregret_pure` and `clogit` independently. The `randregret_pure` command can generate the transformed attributes described in (10). When dealing with a mix of positive and negative attributes, their transformations need to be created in two different runs of `randregret_pure`, that is to say, one for assumed positive attributes and the other for assumed negative attributes. In our case, all the attributes are assumed to have negative signs. Hence, we can create them in a single run of `randregret_pure` using the option `signbeta(neg)`. Finally, given that the function will add the transformed attributes as new Stata variables, the user needs to provide a prefix to name and include them in the dataset. Consequently, we type `prefix(p_)` to declare that all the new attributes' names will start with the `p_` prefix.

```
. randregret_pure tc tt, sign(neg) group(obs) prefix(p_)
. list obs altern choice tt p_tt tc p_tc in 1/3, sepby(obs)
```

	obs	altern	choice	tt	p_tt	tc	p_tc
1.	1	First	0	23	0	6	5
2.	1	Second	0	27	4	4	1
3.	1	Third	1	35	20	3	0

To further illustrate the process of generating the new attributes, we will follow the calculations in (10) to obtain the transformed attribute (`p_tt`) from the original attribute total time (`tt`) for the first choice situation (`obs==1`) of the first individual.

The new transformed attribute `p_tt`, conditional on an assumed negative sign from the original attribute, is given by $x_{i,tt,1}^{\text{PRRM}} = \sum_{j \neq i}^{3} \min(0, x_{j,tt,1} - x_{i,tt,1})$. Subsequently, in matrix format, we will perform the following calculations to obtain `p_tt`.

$$
\begin{aligned}
\mathbf{X}_{tt,1}^{\text{PRRM}} &= \begin{pmatrix} x_{1,tt,1}^{\text{PRRM}} \\ x_{2,tt,1}^{\text{PRRM}} \\ x_{3,tt,1}^{\text{PRRM}} \end{pmatrix} = \begin{pmatrix} \min(0, x_{2,tt,1} - x_{1,tt,1}) + \min(0, x_{3,tt,1} - x_{1,tt,1}) \\ \min(0, x_{1,tt,1} - x_{2,tt,1}) + \min(0, x_{3,tt,1} - x_{2,tt,1}) \\ \min(0, x_{1,tt,1} - x_{3,tt,1}) + \min(0, x_{2,tt,1} - x_{3,tt,1}) \end{pmatrix} \\
&= \begin{pmatrix} \min(0, 27 - 23) + \min(0, 35 - 23) \\ \min(0, 23 - 27) + \min(0, 35 - 27) \\ \min(0, 23 - 35) + \min(0, 27 - 35) \end{pmatrix} \\
&= \begin{pmatrix} 0 + 0 \\ -4 + 0 \\ -12 + -8 \end{pmatrix} = \begin{pmatrix} 0 \\ -4 \\ -20 \end{pmatrix}
\end{aligned}
$$

It is worth mentioning that `randregret_pure` flips the signs of the variables to be used directly in combination with `clogit`. This is because to obtain the choice

probabilities using (11), we need to use the negative of (10), which is exactly what we achieve by invoking `clogit` using the transformed variables. Finally, we can check that we get the same results when running `randregret` using the `rrmfn(pure)` option and using `randregret_pure` together with `clogit`.

```
. clogit choice p_tc p_tt, group(obs) vce(cluster id)
Iteration 0:   log pseudolikelihood = -1132.2901
Iteration 1:   log pseudolikelihood = -1128.3852
Iteration 2:   log pseudolikelihood = -1128.3777
Iteration 3:   log pseudolikelihood = -1128.3777
Conditional (fixed-effects) logistic regression         Number of obs =  3,180
                                                        Wald chi2(2)  =  21.06
                                                        Prob > chi2   = 0.0000
Log pseudolikelihood = -1128.3777                       Pseudo R2     = 0.0310
                                     (Std. err. adjusted for 106 clusters in id)
------------------------------------------------------------------------------
             |               Robust
      choice | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
        p_tc |   -.285628   .0647545    -4.41   0.000    -.4125446   -.1587114
        p_tt |  -.0661575   .0169355    -3.91   0.000    -.0993505   -.0329645
------------------------------------------------------------------------------
```

As we mentioned earlier, `randregret` also allows for the inclusion of ASC for all the models using (12). Below, we run a classic RRM model using the `basealternative(1)` option, which specifies that the first alternative is the reference for the ASC. We list the results here to illustrate the syntax only because the survey was implemented using nonlabeled alternatives.

```
. randregret choice tc tt, group(obs) alternatives(altern) basealternative(1)
> rrmfn(classic) nolog
------------------------------------------------------------------------------
Fitting Classic RRM Model
------------------------------------------------------------------------------

RRM: Classic Random Regret Minimization Model
Case ID variable: obs                           Number of cases    =       1060
Alternative variable: altern                    Number of obs      =       3180
                                                Wald chi2(2)       =      89.98
Log likelihood = -1113.5986                     Prob > chi2        =     0.0000
------------------------------------------------------------------------------
      choice | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
RRM          |
          tc |   -.389129   .0411256    -9.46   0.000    -.4697336   -.3085244
          tt |  -.0910313   .0106063    -8.58   0.000    -.1118192   -.0702433
-------------+----------------------------------------------------------------
ASC          |
       ASC_2 |  -.1673341   .0769052    -2.18   0.030    -.3180656   -.0166026
       ASC_3 |   .0876183   .0815384     1.07   0.283     -.072194    .2474306
------------------------------------------------------------------------------
```

To generate predictions, we can invoke `randregretpred` after running `randregret`. To illustrate this, we rerun the classical RRM model and generate two different predictions. First, using the option `proba`, we generate the `prob` variable, which contains

the predicted probability of (4). Additionally, we also used the `xb` option to generate a variable containing the linear predicted systematic regret of (3).

```
. quietly randregret choice tc tt, group(obs) alternatives(altern)
> rrmfn(classic) noconstant nolog
. randregretpred prob, group(obs) alternatives(altern) proba
. randregretpred xb, group(obs) alternatives(altern) xb
. list obs altern choice id cs tt tc prob xb in 1/12, sepby(obs)
```

	obs	altern	choice	id	cs	tt	tc	prob	xb
1.	1	First	0	1	1	23	6	.22354907	3.4618503
2.	1	Second	0	1	1	27	4	.54655027	2.567855
3.	1	Third	1	1	1	35	3	.22990067	3.4338339
4.	2	First	0	1	2	27	5	.43840211	2.7134208
5.	2	Second	1	1	2	35	4	.19128045	3.5428166
6.	2	Third	0	1	2	23	6	.37031744	2.8821967
7.	3	First	1	1	3	35	3	.25800373	3.2759017
8.	3	Second	0	1	3	23	5	.44187012	2.7378597
9.	3	Third	0	1	3	31	4	.30012616	3.1246728
10.	4	First	0	1	4	27	4	.43840211	2.7134208
11.	4	Second	0	1	4	23	5	.37031744	2.8821967
12.	4	Third	1	1	4	35	3	.19128045	3.5428166

11 Conclusions

We presented the `randregret` command, which allows the user to easily fit four different RRM models, namely, the classic RRM (Chorus 2010), the GRRM (Chorus 2014), the μRRM, and the PRRM (van Cranenburgh, Guevara, and Chorus 2015). We illustrated the results using stated choice discrete choice data in the context of route selection given in van Cranenburgh and Alwosheel (2019). Additionally, we have included additional LR tests that rely on the relationships among the models, which allows us to test whether the data are more likely to be generated by RRM or RUM choice behavior.

12 Acknowledgments

We thank Caspar Chorus, Sander van Cranenburgh, and an anonymous referee for helpful comments and constructive suggestions. We also thank the participants from the 2020 London Stata Conference and 2020 Swiss Stata Users Group meeting for many useful comments and suggestions. Additionally, special thanks from the first author to Valeria Córdova and Mauricio Armijo for their valuable comments on earlier stages of this project. Much of the code, particularly the combination of a Mata evaluator together with the `ml` command, was highly inspired by routines in the book *Maximum Likelihood Estimation with Stata, Fourth Edition* by William Gould, Jeffrey Pitblado, and Brian Poi (2010). Also, many of the data checks performed by `randregret` were greatly inspired by the `mixlogit` command (Hole 2007).

12.1 Funding

This work was produced while Álvaro A. Gutiérrez-Vargas was a PhD student at the Research Centre for Operations Research and Statistics (ORSTAT) at KU Leuven funded by Bijzonder Onderzoeksfonds KU Leuven (Special Research Fund KU Leuven).

12.2 Contribution

Álvaro A. Gutiérrez-Vargas developed the command and drafted the article. Michel Meulders and Martina Vandebroek critically commented on both the article and the command's functionality.

12.3 Conflict of interest

Álvaro A. Gutiérrez-Vargas, Michel Meulders, and Martina Vandebroek declare no conflicts of interest.

13 Programs and supplemental materials

To install a snapshot of the corresponding software files as they existed at the time of publication of this article, type

```
. net sj 21-3
. net install st0649     (to install program files, if available)
. net get st0649         (to install ancillary files, if available)
```

14 References

Bell, D. E. 1982. Regret in decision making under uncertainty. *Operations Research* 30: 961–981. https://doi.org/10.1287/opre.30.5.961.

Buis, M. L. 2007. Stata tip 53: Where did my p-values go? *Stata Journal* 7: 584–586. https://doi.org/10.1177/1536867X0800700408.

———. 2014. Stata tip 116: Where did my p-values go? (Part 3). *Stata Journal* 14: 218–220. https://doi.org/10.1177/1536867X1401400114.

Chorus, C. G. 2010. A new model of random regret minimization. *European Journal of Transport and Infrastructure Research* 10: 181–196. https://doi.org/10.18757/ejtir.2010.10.2.2881.

———. 2014. A generalized random regret minimization model. *Transportation Research, Part B* 68: 224–238. https://doi.org/10.1016/j.trb.2014.06.009.

Chorus, C. G., T. A. Arentze, and H. J. P. Timmermans. 2008. A random regret-minimization model of travel choice. *Transportation Research, Part B* 42: 1–18. https://doi.org/10.1016/j.trb.2007.05.004.

Chorus, C. G., and M. Bierlaire. 2013. An empirical comparison of travel choice models that capture preferences for compromise alternatives. *Transportation* 40: 549–562. https://doi.org/10.1007/s11116-012-9444-3.

Chorus, C. G., J. M. Rose, and D. A. Hensher. 2013. Regret minimization or utility maximization: It depends on the attribute. *Environment and Planning B: Planning and Design* 40: 154–169. https://doi.org/10.1068/b38092.

Gould, W., J. Pitblado, and B. Poi. 2010. *Maximum Likelihood Estimation with Stata*. 4th ed. College Station, TX: Stata Press.

Gutierrez, R. G., S. Carter, and D. M. Drukker. 2001. sg160: On boundary-value likelihood-ratio tests. *Stata Technical Bulletin* 60: 15–18. Reprinted in *Stata Technical Bulletin Reprints*. Vol. 10, pp. 269–273. College Station, TX: Stata Press.

Hole, A. R. 2007. Fitting mixed logit models by using maximum simulated likelihood. *Stata Journal* 7: 388–401. https://doi.org/10.1177/1536867X0700700306.

Loomes, G., and R. Sugden. 1982. Regret theory: An alternative theory of rational choice under uncertainty. *Economic Journal* 92: 805–824. https://doi.org/10.2307/2232669.

Manski, C. F. 1977. The structure of random utility models. *Theory and Decision* 8: 229–254. https://doi.org/10.1007/BF00133443.

Molenberghs, G., and G. Verbeke. 2007. Likelihood ratio, score, and Wald tests in a constrained parameter space. *American Statistician* 61: 22–27. https://doi.org/10.1198/000313007X171322.

Self, S. G., and K.-Y. Liang. 1987. Asymptotic properties of maximum likelihood estimators and likelihood-ratio tests under nonstandard conditions. *Journal of the American Statistical Association* 82: 605–610. https://doi.org/10.2307/2289471.

Train, K. E. 2009. *Discrete Choice Methods with Simulation*. 2nd ed. Cambridge: Cambridge University Press.

van Cranenburgh, S. 2018. Small value-of-time experiment, Netherlands. 4TU.Centre for Research Data, Dataset. https://doi.org/10.4121/uuid:1ccca375-68ca-4cb6-8fc0-926712f50404.

van Cranenburgh, S., and A. Alwosheel. 2019. An artificial neural network based approach to investigate travellers' decision rules. *Transportation Research, Part C* 98: 152–166. https://doi.org/10.1016/j.trc.2018.11.014.

van Cranenburgh, S., C. A. Guevara, and C. G. Chorus. 2015. New insights on random regret minimization models. *Transportation Research, Part A* 74: 91–109. https://doi.org/10.1016/j.tra.2015.01.008.

van Cranenburgh, S., and C. G. Prato. 2016. On the robustness of random regret minimization modelling outcomes towards omitted attributes. *Journal of Choice Modelling* 18: 51–70. https://doi.org/10.1016/j.jocm.2016.04.004.

About the authors

Álvaro A. Gutiérrez-Vargas is a PhD student at the Research Centre of Operation Research and Statistics (ORSTAT) at KU Leuven in Belgium. He earned a Bachelor of Science in economics from the University of Chile. His research interests are mainly methodological and focused on computational statistics and discrete choice models.

Michel Meulders is an associate professor at the Faculty of Economics and Business at KU Leuven in Belgium. He earned a PhD in psychology from KU Leuven. His research is mainly methodological and focuses on modeling choice behavior and three-way data. He has been published in such journals as *Journal of Statistical Software*, *Psychometrika*, *Journal of Educational and Behavioural Statistics*, and *British Journal of Mathematical and Statistical Psychology*.

Martina Vandebroek is a full professor at the Faculty of Economics and Business at KU Leuven in Belgium. She earned a PhD in actuarial sciences from KU Leuven. She is interested in the design of experiments, discrete choice experiments, and multivariate statistics. She has been published in *Transportation Research B*, *Journal of Choice Modelling*, *Marketing Science*, and *Journal of Statistical Software*, among other journals.

A Technical appendix

A.1 Generic score functions for RRM models

Without loss of generality, we can state that the log likelihood of the four RRM models presented in this article can be represented by (16). In particular, when R^*_{in} is replaced by (3), (5), (7), or (9), we can fit, respectively, the classical RRM, the GRRM, the μRRM, or the PRRM model.

$$\begin{aligned}\ln L &= \sum_{n=1}^{N}\sum_{i=1}^{J} y_{in} \ln\left(P^*_{in}\right) \\ &= \sum_{n=1}^{N}\sum_{i=1}^{J} y_{in} \ln\left\{\frac{\exp\left(-R^*_{in}\right)}{\sum_{j=1}^{J}\exp\left(-R^*_{jn}\right)}\right\} \\ &= -\sum_{n=1}^{N}\sum_{i=1}^{J} y_{in} R^*_{in} - \sum_{n=1}^{N}\sum_{i=1}^{J} y_{in} \ln\left\{\sum_{j=1}^{J}\exp\left(-R^*_{jn}\right)\right\}\end{aligned} \tag{16}$$

Furthermore, any partial derivative of the log likelihood with respect to any parameter $\theta \in \boldsymbol{\theta}$, where $\boldsymbol{\theta}$ stands for the full set of parameters of the model, can be expressed as in (17). The rank of $\boldsymbol{\theta}$ will depend on the particular model.

$$\begin{aligned}\frac{\partial \ln L}{\partial \theta} &= -\sum_{n=1}^{N}\sum_{i=1}^{J} y_{in}\frac{\partial R^*_{in}}{\partial \theta} + \sum_{n=1}^{N}\sum_{i=1}^{J} y_{in}\left(\sum_{j=1}^{J} P_{jn}\frac{\partial R^*_{jn}}{\partial \theta}\right) \\ &= -\sum_{n=1}^{N}\sum_{i=1}^{J}\left(y_{in} - P_{in}\right)\left(\frac{\partial R^*_{in}}{\partial \theta}\right)\end{aligned} \tag{17}$$

In the appendices A.2–A.5, we will list the partial derivatives, also known as score functions, per type of parameter in each type of model. Additionally, note that, in any case, we can check that $\partial R^*_{in}/\partial\alpha_i = 1$, where α_i represents the coefficient associated with the ASC of alternative i.

A.2 Score functions for the classical RRM model

To obtain the log likelihood of the classic RRM model, we need to substitute R^*_{in} in (16) by (3). Accordingly, the set of parameters $\boldsymbol{\theta}$ is now given by $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\alpha})'$. Here $\boldsymbol{\beta}$ is an $m \times 1$ vector of alternative-specific regression coefficients, and $\boldsymbol{\alpha}$ is a $(J-1) \times 1$ vector of ASC. Subsequently, the score functions of the classical RRM model will be described as follows:

$$\begin{aligned}\frac{\partial \ln L}{\partial \boldsymbol{\theta}} &= \left(\frac{\partial \ln L}{\partial \beta_1}, \ldots, \frac{\partial \ln L}{\partial \beta_M}, \frac{\partial \ln L}{\partial \alpha_1}, \ldots, \frac{\partial \ln L}{\partial \alpha_{J-1}}\right) \\ &= \left(\frac{\partial \ln L}{\partial \boldsymbol{\beta}}, \frac{\partial \ln L}{\partial \boldsymbol{\alpha}}\right)\end{aligned}$$

Finally, to obtain the expression for $\partial \ln L/\partial\beta_m$, we need to replace (18) into (17).

$$\frac{\partial R_{in}}{\partial \beta_m} = \sum_{j \neq i}^{J} \left[\frac{\exp\left\{\beta_m \left(x_{jmn} - x_{imn}\right)\right\} \times \left(x_{jmn} - x_{imn}\right)}{1 + \exp\left\{\beta_m \left(x_{jmn} - x_{imn}\right)\right\}}\right] \tag{18}$$

A.3 Score functions for the GRRM model

The log likelihood of the GRRM model can be constructed by replacing the term R^*_{in} in (16) with (5). Hence, the full set of parameters $\boldsymbol{\theta}$ is now given by $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\alpha}, \gamma^*)'$. Here $\boldsymbol{\beta}$ is an $m \times 1$ vector of alternative-specific regression coefficients, $\boldsymbol{\alpha}$ is a $(J-1) \times 1$ vector of ASC, and γ^* is a scalar equal to the parameter γ in the logit scale. Hence, the corresponding score functions are described by

$$\begin{aligned}\frac{\partial \ln L}{\partial \boldsymbol{\theta}} &= \left(\frac{\partial \ln L}{\partial \beta_1}, \ldots, \frac{\partial \ln L}{\partial \beta_M}, \frac{\partial \ln L}{\partial \alpha_1}, \ldots, \frac{\partial \ln L}{\partial \alpha_{J-1}}, \frac{\partial \ln L}{\partial \gamma^*}\right) \\ &= \left(\frac{\partial \ln L}{\partial \boldsymbol{\beta}}, \frac{\partial \ln L}{\partial \boldsymbol{\alpha}}, \frac{\partial \ln L}{\partial \gamma^*}\right)\end{aligned}$$

Additionally, to obtain the expression for $\partial \ln L/\partial\beta_m$, we need to replace (19) into (17).

$$\frac{\partial R^{\text{GRRM}}_{in}}{\partial \beta_m} = \sum_{j \neq i}^{J} \left[\frac{\exp\left\{\beta_m \left(x_{jmn} - x_{imn}\right)\right\} \times \left(x_{jmn} - x_{imn}\right)}{\gamma + \exp\left\{\beta_m \left(x_{jmn} - x_{imn}\right)\right\}}\right] \tag{19}$$

However, the score function of the parameter γ^* needs a slightly different treatment. As mentioned earlier, the optimization procedure does not directly fit the parameter γ;

instead, it fits the model by using an ancillary parameter: $\gamma^* = \text{logit}(\gamma)$. Hence, we model the parameter γ in the logit scale. This fact has a direct impact on the score function of parameter γ^*. Using the chain rule, we can state

$$\frac{\partial \ln L}{\partial \gamma} = \frac{\partial \ln L}{\partial \gamma^*} \times \frac{\partial \gamma^*}{\partial \gamma}$$

Subsequently, solving $\partial\gamma^*/\partial\gamma$ and rearranging terms, we see in (20) that to compute the score function of the parameter γ^*, we need to adjust the partial derivative from the log likelihood with respect to γ by a factor of $\gamma(1-\gamma)$.

$$\frac{\partial \ln L}{\partial \gamma^*} = \frac{\partial \ln L}{\partial \gamma} \times \gamma(1-\gamma) \tag{20}$$

The expression for $\partial \ln L/\partial\gamma$ can be computed by replacing (21) into (17), which together with (20) gives us the required expression for $\partial \ln L/\partial\gamma^*$.

$$\frac{\partial R_{in}^{\text{GRRM}}}{\partial \gamma} = \sum_{j\neq i}^{J} \sum_{m=1}^{M} \left[\frac{1}{\gamma + \exp\left\{\beta_m \left(x_{jmn} - x_{imn}\right)\right\}} \right] \tag{21}$$

A.4 Score functions for μRRM model

The μRRM model has a log likelihood that is a particular case of (17), where R_{in}^* is replaced by (7). Thus, the full set of parameters $\boldsymbol{\theta}$ is now described by $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\alpha}, \mu^*)'$. Here $\boldsymbol{\beta}$ is an $m \times 1$ vector of alternative-specific regression coefficients, $\boldsymbol{\alpha}$ is a $(J-1)\times 1$ vector of ASC, and μ^* is a scalar equal to the μ parameter in a transformed scale. Thus, the corresponding score functions can be represented by

$$\begin{aligned}\frac{\partial \ln L}{\partial \boldsymbol{\theta}} &= \left(\frac{\partial \ln L}{\partial \beta_1}, \ldots, \frac{\partial \ln L}{\partial \beta_M}, \frac{\partial \ln L}{\partial \alpha_1}, \ldots, \frac{\partial \ln L}{\partial \alpha_{J-1}}, \frac{\partial \ln L}{\partial \mu^*} \right) \\ &= \left(\frac{\partial \ln L}{\partial \boldsymbol{\beta}}, \frac{\partial \ln L}{\partial \boldsymbol{\alpha}}, \frac{\partial \ln L}{\partial \mu^*} \right)\end{aligned}$$

First, by replacing (22) back into (17), we can easily obtain the expression for $\partial \ln L/\partial\beta_m$.

$$\frac{\partial R_{in}^{\mu\text{RRM}}}{\partial \beta_m} = \sum_{j\neq i}^{J} \left(\frac{\exp\left\{(\beta_m/\mu) \times (x_{jmn} - x_{imn})\right\} \times (x_{jmn} - x_{imn})}{\mu \times \left[1 + \exp\left\{(\beta_m/\mu) \times (x_{jmn} - x_{imn})\right\}\right]} \right) \tag{22}$$

The μRRM model, similarly to the GRRM model, also fits the parameter μ by using an unbounded ancillary parameter: $\mu^* = \ln\{\mu/(M-\mu)\}$. Accordingly, this transformation needs to be accounted for when computing the score function of the parameter μ^*. Using the chain rule, we can state

$$\frac{\partial \ln L}{\partial \mu} = \frac{\partial \ln L}{\partial \mu^*} \times \frac{\partial \mu^*}{\partial \mu}$$

Solving for $\partial\mu^*/\partial\mu$ and rearranging terms, we can see that the score function of the parameter μ^* is the same as the partial derivative of the log likelihood with respect to μ multiplied by a factor equal to $\mu\,(M-\mu)\,/M$.

$$\frac{\partial \ln L}{\partial \mu^*} = \frac{\partial \ln L}{\partial \mu} \times \frac{\mu\,(M-\mu)}{M} \tag{23}$$

Finally, the expression for $\partial \ln L/\partial\mu$ can be obtained by replacing (24) and (25) into (17), which together with (23) provides the required expression for $\partial \ln L/\partial\mu^*$.

$$\frac{\partial R_{in}^{\mu\text{RRM}}}{\partial \mu} = \sum_{j\neq i}^{J}\sum_{m=1}^{M} R_{i\leftrightarrow j,m}^{\mu\text{RRM}} + \mu \times \sum_{j\neq i}^{J}\sum_{m=1}^{M} \frac{\partial R_{i\leftrightarrow j,m}^{\mu\text{RRM}}}{\partial \mu} \tag{24}$$

$$\frac{\partial R_{i\leftrightarrow j,m}^{\mu\text{RRM}}}{\partial \mu} = \left(\frac{\exp\left\{(\beta_m/\mu)\times(x_{jmn}-x_{imn})\right\}\times(x_{jmn}-x_{imn})\times\beta_m}{\mu^2\times\left[1+\exp\left\{(\beta_m/\mu)\times(x_{jmn}-x_{imn})\right\}\right]} \right) \tag{25}$$

A.5 Score functions for the PRRM model

We can recover the log likelihood of the PRRM model by replacing the expression R_{in}^* in (16) by (9). Thus, the full set of parameters $\boldsymbol{\theta}$ is now described by $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\alpha})'$. Here $\boldsymbol{\beta}$ is an $m \times 1$ vector of alternative-specific regression coefficients, and $\boldsymbol{\alpha}$ is a $(J-1)\times 1$ vector of ASC. Consequently, the score functions are then

$$\begin{aligned}\frac{\partial \ln L}{\partial \boldsymbol{\theta}} &= \left(\frac{\partial \ln L}{\partial \beta_1}, \ldots, \frac{\partial \ln L}{\partial \beta_M}, \frac{\partial \ln L}{\partial \alpha_1}, \ldots, \frac{\partial \ln L}{\partial \alpha_{J-1}} \right) \\ &= \left(\frac{\partial \ln L}{\partial \boldsymbol{\beta}}, \frac{\partial \ln L}{\partial \boldsymbol{\alpha}} \right)\end{aligned}$$

Accordingly, we can obtain the expression for $\partial \ln L/\partial\beta_m$ by replacing (26) into (17).

$$\frac{\partial R_{in}^{\text{PURE}}}{\partial \beta_m} = x_{imn}^{\text{PURE}} \tag{26}$$