Method Dissemination Articles

Estimating Item Parameters in Multistage Designs With the tmt Package in R

Jan Steinfeld1,2, Alexander Robitzsch3,4


Quantitative and Computational Methods in Behavioral Sciences, 2023, Article e10087, https://doi.org/10.5964/qcmb.10087

Received: 2022-08-18 • Accepted: 2023-03-29 • Published (VoR): 2023-11-06

Handling Editor: Ross Jacobucci, University of Notre Dame, Notre Dame, IN, USA

Corresponding Author: Jan Steinfeld, Department of Developmental and Educational Psychology, Faculty of Psychology, University of Vienna, Liebiggasse 5, A-1010 Vienna, Austria. E-mail: jan.d.steinfeld@gmail.com

Supplementary Materials: Code [see Index of Supplementary Materials]

Abstract

Various likelihood-based methods are available for the parameter estimation of item response theory models (IRT), leading to comparable parameter estimates. Considering multistage testing (MST) designs, Glas (1988; https://doi.org/10.2307/1164950) stated that the conditional maximum likelihood (CML) method in its original formulation leads to severely biased parameter estimates. A modified CML estimation method for MST designs proposed by Zwitser and Maris (2015; https://doi.org/10.1007/s11336-013-9369-6) finally provides asymptotically unbiased item parameter estimates. Steinfeld and Robitzsch (2021b; https://doi.org/10.31234/osf.io/ew27f) complemented this method to MST designs with probabilistic routing strategies. For both proposed modifications additional software solutions are required since design-specific information must be incorporated into the estimation process. An R package that has implemented both modifications is "tmt". In this article, first, the proposed solutions of the CML estimation in MST designs are illustrated, followed by the main part, the demonstration of the CML item parameter estimation with the R package "tmt". The demonstration includes the process of model specification, data simulation, and item parameter estimation, considering two different routing types of deterministic and probabilistic MST designs.
For several years now various international large-scale assessments (ILSA) transitioned from paper-based to computer-based assessments (e.g., Brennan, 2006). Some ILSA thereby also successfully applied adaptive test designs (e.g., Chang, 2015). Among these ILSAs are several well-known programs like the Programme for International Student Assessment (PISA; OECD, 2019a, 2020), and the Programme for the International Assessment of Adult Competencies (PIAAC; OECD, 2019b). Adaptive test designs can be roughly split into computerized adaptive tests (CAT; Lord, 1971a, 1980; Owen, 1975; van der Linden & Glas, 2010; Wainer et al., 2000; Weiss, 1976, 1983) with test administration on item level and multistage tests (MST; Angoff & Huddleston, 1958; Lord, 1968, 1971b; Lord et al., 1968; Luecht & Nungester, 1998; Zenisky et al., 2009) where pre-specified groups of items are selected in the administration process. The application of adaptive testing has become an essential testing method (e.g., Chen et al., 2014; Dean & Martineau, 2012) used in the mentioned ILSAs and other areas such as psychological assessment (e.g., Kubinger & Holocher-Ertl, 2014), or classroom assessments (Chang, 2015). Adaptive test designs have in common that these are usually more efficient in terms of shorter test lengths while providing equal or even higher measurement precision. Furthermore, this type of design is associated with higher predictive validity compared to linear fixed-length tests (Betz & Weiss, 1974; Chang, 2015; Cronbach & Gleser, 1957; Hendrickson, 2007; Jodoin et al., 2006; Kim & Plake, 1993; Linn et al., 1969; Lord, 1980; Schnipke & Reese, 1997; Wainer et al., 2000; Weiss, 1982; Weiss & Kingsbury, 1984). In particular, the advantages of adaptive test designs will occur for the more extreme abilities at the lower and upper end of the measurement scale (Hendrickson, 2007; Lord, 1974, 1980).

As already stated, adaptive test designs can be split based on modalities of item selection methods into item-by-item designs (here referred to CAT) and those where pre-assembled groups of items are administered (here referred to MST). While the item-by-item designs are bound to the computer due to the requirement of constantly estimation of person parameters for the item-selection, MST designs can also be administered in paper-pencil forms (see, e.g., Cronbach & Gleser, 1957; Kubinger & Holocher-Ertl, 2014; Linn et al., 1969). Some contributions do not separate these two designs so strictly. As emphasized by Chang (2015), for example, both designs could be regarded as sequential designs (see also Han & Guo, 2014; Kaplan & de la Torre, 2020; Luo & Wang, 2019; Zheng & Chang, 2014, for dynamic multistage designs).

The present work refers to the item parameter estimation with the conditional maximum likelihood method under the application of MST design. As stated by Glas (1988) the common CML approach item parameter estimates are severely biased and only feasible by a modification of the common CML approach proposed by Zwitser and Maris (2015). With the proposed modification, the application of conventional software
and R packages that have implemented the CML method for item parameter estimating is not applicable. Therefore, the package tmt was developed. Considering adaptive test designs, normally the step of item parameter estimation is done before the preparation of the test design. Referring to MST designs in ILSA, provisional item difficulties are applied for the test construction, the actual item parameter estimation is carried out afterward. In PIAAC and PISA, data were collected using an adaptive MST design to subsequently estimate item parameters (OECD, 2019a, 2019b).

Other situations of posterior item parameter estimation could be a rescaling of already established MST designs. This might be, e.g., that a test designed, calibrated, and administered in one educational entity (district, state, etc.) should be ported to a new entity, but requires item parameters be calibrated to the local student population.

**Multistage Testing and Routing Strategies**

In MST designs, subsequent modules are selected based on the test persons’ performance in the actual module. Modules are collections of items covering certain statistical characteristics like mean item difficulties and variances of the item difficulties within modules. In addition, some relevant factors might also be non-statistical like comparable word count, item types, the balance of answer keys, and balance of the item contents represent specific competencies or domains to test within and across modules (Magis et al., 2017; see also the relation to testlets; Lord, 1980; Wainer & Kiely, 1987) might be also relevant factors. Each module in the routing process is referred to as a stage in the MST design. The combination of several processed modules among stages is called a path (see Figure 1 for an example).

**Figure 1**
*Example of a Multistage Design With Seven Modules, Three Stages, and Four Paths*
Tests with MST designs usually start with a module with a comparatively wide spectrum of item difficulties. If several modules are available the best suited module is selected based on additional pre-information regarding the person’s ability (sometimes the first module is also selected randomly, e.g., to achieve particular representativeness expectations). Based on the performance in this routing module, additional modules are administered, which are best suited given the currently estimated ability. The process of receiving additional modules is called routing (Yan, Lewis, et al., 2014).

Defining rules for the selection of modules is a key factor in MST, as it is linked to efficiency and might also impact the precision of item parameter estimation (Lord, 1980; Yan, Lewis, et al., 2014). In the following, the introduced routing strategies are categorized into deterministic and probabilistic routing. In deterministic routing all persons with the same performance (same raw score) in the same module \( m_h[b] \) of \( B \) modules with \( b = 1, ..., B \) in the same stage of \( H \) stages with \( h = 1, ..., H \) are routed to the same subsequent module\(^1\). Several deterministic routing strategies are conceivable. A common routing strategy –number-correct score (NC)– refers to the number of solved items in the current module. A person \( p = 1, ..., P \) with ability \( \theta_p \) and raw score \( x_{p+b} = \sum_{i \in m^b_p} x_{pi} \) in module \( m^b \), will be routed to an easier module if \( x_{p+b} \leq c^b \) and in the remaining cases to a more difficult module (see also Lord, 1980; Zenisky et al., 2009). Another deterministic routing strategy based on the NC score is the incorporation of the information of all modules processed by person \( p \) and referred to as the cumulative number-correct score (cNC; Kim et al., 2015; Svetina et al., 2019). Here, the number of solved items in the current module is added to the number of solved items in the previously processed modules (if applicable), so that information from all processed modules is used for further routing. Compared to sequential routing, more information about the person’s ability is gained in the routing process, and therefore, a more valid routing might be possible. Another approach of deterministic routing is incorporating the specific item difficulties instead of the raw score of that module. This is referred to as item response theory (IRT)-based routing in the literature (Yan, von Davier, et al., 2014).

Probabilistic routing, first introduced in PIAAC (Chen et al., 2014; Yamamoto & Khorramdel, 2018; Yamamoto et al., 2018), is characterized by additional predetermined probabilities which form, together with the introduced deterministic routing rule, the routing decision. Here, persons with the same performance \( x_{p+b} \) in the same module \( m^b \) are only routed with probability \( p^b(x_{p+b}) \) to the optimal module \( m^{b+1} \) and in the remaining case to another easier or more difficult module. The probability of routing to the optimal module increases with increasing or decreasing NC score. This type of routing safeguards a minimal item exposure rate.

---

1) For the illustration used here, it is not necessary to differentiate the module assignment into stages, since no module was assigned to multiple stages. For these reasons and the associated improvement of readability of the equations, the index \( h \) for stages is dropped in the following.
Controlling the item exposure rates for the interested population(s) is a key factor for the subsequent item parameter estimation. Particularly with ILSA, which is applied in many countries, different languages, and educational backgrounds, and designed for different achievement groups, the item exposure control is critical. In PIAAC e.g., routing probabilities were determined on expected item exposure rates for each interested subpopulation by educational level and skills (Chen et al., 2014). As stated by Rutkowski et al. (2022), applying a probabilistic routing strategy is also promising to reduce bias and increase the precision of both item and person parameter estimation across highly varied achievement distributions across countries in ILSAs.

As stated, the selected routing strategy moderates the efficiency of the MST. By comparing for example different deterministic routing approaches, it was concluded by Svetina et al. (2019) that IRT-based routing performs best. However, the simpler to-implement NC-based routing strategy does not perform significantly worse considering the median of person parameter recovery rates, as Svetina et al. (2019) stated.

**Parameter Estimation**

Several methods for calibrating item parameters with data obtained by MST designs are conceivable. The item parameters are often regarded as fixed, and the persons are treated as either fixed or random (see, e.g., De Boeck, 2008; Holland, 1990; Lord et al., 1968; Molenaar, 1995b; San Martin & De Boeck, 2015, for further discussion on this topic).

In the following solely dichotomous item responses are considered utilizing the Rasch model (RM; Rasch, 1960) and the conditional maximum likelihood (CML; Andersen, 1972, 1973) estimation method. Other estimation approaches are available, in particular, marginal maximum likelihood estimation (MML; Bock & Aitkin, 1981; Bock & Lieberman, 1970; Thissen, 1982) with the assumption of normal or a non-normal trait distribution (Xu & von Davier, 2008) or Bayesian estimation methods (see, e.g., Draxler, 2018; Fox, 2010; Levy & Mislevy, 2017; Rupp et al., 2004).

**Estimation Approaches in MST Designs**

Regarding the scaling of data obtained by an MST design, the MML estimation approach can be applied without any further special treatment concerning MST designs (see, e.g., Eggen & Verhelst, 2011; Glas, 1988; Wang et al., 2020). This is different for the CML estimation method, which is only feasible by modifying the common CML approach as proposed by Zwitser and Maris (2015). Supporters of the CML estimation approach might highlight its superiority because this type of item parameter estimation is independent of assumptions of the trait distribution (Eggen & Verhelst, 2011; Glas, 1988; Kubinger et al., 2012; Zwitser & Maris, 2015), since no distribution assumption for the person parameters is required. It is also often emphasized that the CML estimation comes close to the idea of person-free assessment (Molenaar, 1995a) required for postulating specific objectivity (Rasch, 1967, 1977). Steinfeld and Robitzsch (2021a) studied different
estimation approaches for MST, considering different MST designs and trait distributions in a simulation study. Their results indicated that in cases of substantial violation of the normal distribution, the MML approach assuming that traits are normally distributed led to relatively large RMSE compared to the modified CML estimation method (see also Casabianca, 2011; Casabianca & Lewis, 2015; Holland & Thayer, 2000; Xu & von Davier, 2008). From a more theoretical perspective, however, it should be noted that as the number of items increases \((I → ∞)\), the theoretically specified distribution for \(θ\) is meaningless as an empirical prior. For very long tests the specified distribution has therefore no meaningful influence (cf. also Clarke & Junker, 1991; Cliff & Donoghue, 1992; Douglas, 1997; Douglas, 2001; Ellis & Junker, 1997; Junker, 1993; Kiefer & Wolfowitz, 1956; Peress, 2012; Strout, 1990).

**Conditional Maximum Likelihood Estimation Method**

As stated, in the following dichotomous item responses and the RM are considered. Let \(X_{pi}\) denotes an independent distributed random response variable with the realization \(x_{pi} = 1\) if person \(p\) solves item \(i\) and \(x_{pi} = 0\) otherwise. The probability of solving item \(i = 1, ..., I\) with difficulty \(β_i\) by person \(p = 1, ..., P\) with ability \(θ_p\) can be expressed as

\[
P(X_{pi} = x_{pi} | θ_p, β_i) = \frac{\exp[x_{pi}(θ_p − β_i)]}{1 + \exp(θ_p − β_i)},
\]

with \(x_{pi} = 1\). The person-specific likelihood \(L(x_p | θ_p, β)\) with responses \(x_p = (x_{p1}, x_{p2}, ..., x_{pi})\) of person \(p\) with ability \(θ_p\), item difficulties \(β\) and assumed local independence is proportional to

\[
L(x_p | θ_p, β) = \frac{\exp(x_{p+}θ_p − \sum_{i=1}^{I} x_{pi}β_i)}{\prod_{i=1}^{I}[1 + \exp(θ_p − β_i)]}
\]

where \(x_{p+} = \sum_{i=1}^{I} x_{pi}\) denotes the raw score of person \(p\). In the following, we will omit the person index \(p\) in \(x_{p+}\). As stated, one approach estimating the item parameters is the CML estimation method. Applying the CML method, conditional likelihoods are used for the estimation. By conditioning on the raw scores of the persons (person marginal sums), which is also referred to as minimal sufficient statistic for person parameter \(θ_p\) (Andersen, 1972, 1973; Fischer, 1974; Rasch, 1960), the person parameter \(θ\) is canceled. For a more detailed depiction of the CML method see for instance Fischer (2007). The likelihood for the response matrix \(X\) in the CML case with \(s_i = \sum_{p=1}^{P} x_{pi}\) as item score of item \(i\), \(n_{x_i}\) as the number of persons with raw score \(\sum_{i} x_i = x_+\) results in Equation 6. Here the crucial part of the estimation is the calculation of the elementary symmetric function (ESF) \(γ(x_+, β)\) of order \(x_+\) and \(β_1, β_2, ..., β_I\).
\[ L(X \mid \theta, \beta) = \frac{\exp(\sum_{p=1}^{P} x_p \theta_p - \sum_{i=1}^{l} x_p \beta_i)}{\prod_{p=1}^{P} \prod_{l=1}^{l_i} [1 + \exp(\theta_p - \beta_i)]} \]
\[ = \frac{\exp(\sum_{p=1}^{P} x_p \theta_p - \sum_{i=1}^{l} s_i \beta_i)}{\prod_{p=1}^{P} \prod_{l=1}^{l_i} [1 + \exp(\theta_p - \beta_i)]} \]
\[ L(x_+ \mid \beta) = \frac{\exp(\sum_{p=1}^{P} x_p \theta_p) \prod_{p=1}^{P} \sum_{x_p} \exp(-\sum_{i=1}^{l} x_p \beta_i)}{\prod_{p=1}^{P} \prod_{l=1}^{l_i} [1 + \exp(\theta_p - \beta_i)]} \]
\[ L(X \mid x_+, \beta) = \frac{L(X \mid \theta, \beta)}{L(x_+ \mid \beta)} \]
\[ = \frac{\exp(-\sum_{i=1}^{l} s_i \beta_i)}{\prod_{p=1}^{P} \sum_{x_p} \exp(-\sum_{i=1}^{l} x_p \beta_i)} \]
\[ L_{CML}(X \mid x_+, \beta) \propto \frac{\exp(-\sum_{i=1}^{l} s_i \beta_i)}{\prod_{x_i=0}^{x_i=x} \gamma(x_+, \beta)^{x_i}} \]

In total there are \( \binom{l}{x_+} \) different possibilities for obtaining the score \( x_+ \). Summing over these different possibilities \( \binom{l}{x_+} \) is described by the equation

\[ \gamma(x_+, \beta) = \sum_{\{x_i \mid \sum x_i = x_+\}} \exp(-\sum_{i=1}^{l} x_i \beta_i). \]

The calculation of the ESF can become a bottleneck in the estimation process, particularly with larger amounts of items. Therefore, several methods to compute the ESF have been proposed, which differ mainly in accuracy and speed (see, e.g., Formann, 1986; Liou, 1994; Verhelst et al., 1984). Molenaar (1995b) stated that the resulting estimates of \( \hat{\beta} \) by maximizing Equation (6) are consistent, asymptotically efficient, and asymptotically normally distributed.

**CML Estimation in MST Designs**

As illustrated in the Introduction, as well as in the Multistage Testing and Routing Strategies section, in MST designs, persons obtain additional modules based on their performances and pre-specified routing rules. Persons with higher scores in the same modules are usually routed to more difficult modules, and persons with lower scores usually to easier modules.

The application of deterministic routing rules causes that not all raw scores are possible in each path of the design, as would be the case in a linear test administration. Suppose the routing from module \( m^{[1]} \) to module \( m^{[2]} \) with six items in each module
based on the deterministic routing rule, that the raw score in module \( m^{[1]} \) is greater than three. This results in the fact that only raw scores greater than three to a maximum of twelve can be observed in the path \( m^{[1, 2]} \), but not the raw scores zero, one, two, or three. This deviates from expectations in the calculation of the common ESF. Therefore, the common CML item parameter estimation leads to severely biased item parameter estimates (Glas, 1988; see also Eggen & Verhelst, 2011; Kubinger et al., 2012).

CML Estimation in Deterministic MST Designs

Zwitser and Maris (2015) tackled the issue of CML estimation for deterministic routing due to considering the respective MST design in the CML estimation process. They proposed a modification of the symmetric function to consider only those raw scores which can occur due to the specific MST design. They demonstrated that the resulting item parameter estimates are consistent with this modification.

Referring to their solution, a person \( p \) with raw score \( x_p \) is routed in the deterministic case from module \( m^{[b]} \) to the next module based on a cut score \( c^{[b]} \). Therefore, the probability of score \( x^{[1, 2]} \) in the two modules \( m^{[1, 2]} \) with a given ability \( \theta \) and the condition that the raw score in the first module \( m^{[1]} \) is not greater than the cut score \( c^{[1]} \) and \( P(X^{[1]}_+ \leq c^{[1]} | \theta) \), can be expressed as

\[
P_{m^{[1, 2]}}(x^{[1, 2]} | \theta, X^{[1]}_+ \leq c^{[1]} | \theta) = \frac{P_{m^{[1, 2]}}(x^{[1, 2]} | \theta)}{P_{m^{[1, 2]}}(X^{[1]}_+ \leq c^{[1]} | \theta)}.
\]

(8)

Note that \( P_{m^{[1, 2]}}(X^{[1]}_+ \leq c^{[1]} | X^{[1, 2]}_+, \theta) \) equals one since the condition implies the inequality.

The distribution of \( X^{[1]}_+ \) and \( X^{[2]}_+ \) conditioned on a raw score of \( x^{[1, 2]}_+ \), can be expressed with the common CML approach as follows

\[
P_{m^{[1, 2]}}(x^{[1, 2]}_+ | x^{[1, 2]}_+) = \frac{\prod_{i=1}^{d^{[1]}} \exp(-x^{[1]}_i \beta^{[1]}) \prod_{j=1}^{d^{[2]}} \exp(-x^{[2]}_j \beta^{[2]})}{\sum_{j=0}^{I^{[2]}} \gamma^{[1]}_j(m^{[1]}) \gamma^{[x^{[1, 2]}_+ - j]}(m^{[2]})}.
\]

(9)

The probability of \( X^{[1]}_+ \) lower or equal of cut score \( c^{[1]} \) conditioned on \( x^{[1, 2]}_+ \) is

\[
P_{m^{[1, 2]}}(X^{[1]}_+ \leq c^{[1]} | x^{[1, 2]}_+) = \frac{\sum_{j=0}^{d^{[1]}} Y_j(m^{[1]}) \gamma^{[x^{[1, 2]}_+ - j]}(m^{[2]})}{\sum_{j=0}^{d^{[2]}} Y_j(m^{[1]}) \gamma^{[x^{[1, 2]}_+ - j]}(m^{[2]})}.
\]

(10)

The probability for a raw score of \( x^{[1, 2]}_+ \) conditioned on score \( x^{[1, 2]}_+ \) reached in both modules \( m^{[1, 2]} \) under the condition that the raw score \( x^{[1]}_+ \) in module \( m^{[1]} \) smaller or equal to the previously defined cut score \( c^{[1]} \), can be described with the two Equations (9) and (10) as follows
\[
P_{m^{[1,2]} | x^{[1,2]}, X^+} \leq c^{[1]} = \frac{P_{m^{[1,2]}(x^{[1,2]}, X^+_1 \leq c^{[1]} | x^{[1,2]})}}{P_{m^{[1,2]}(X^+_1 \leq c^{[1]} | x^{[1,2]})}} = \frac{P_{m^{[1,2]}(x^{[1,2]} | x^{[1,2]})}}{P_{m^{[1,2]}(X^+_1 \leq c^{[1]} | x^{[1,2]})}} (11)
\]

The probability 
\[
P_{m^{[1,2]}(X^+_1 \in C^{[1]} | x^{[1,2]})} = \frac{\sum^{j=0}_{j=0} P^{[1]}_j Y_j(m^{[1]} | y_{x^{[1,2]} - j}(m^{[2]})} \text{.} \]

The probability 
\[
P_{m^{[1,2]}(x^{[1,2]}, X^+_1 \in C^{[1]})} \text{ can be described by Equation 9 and Equation 12 as follows}
\]

\[
P_{m^{[1,2]}(x^{[1,2]}, X^+_1 \in C^{[1]})} = \frac{P_{m^{[1,2]}(x^{[1,2]}, X^+_1 \in C^{[1]} | x^{[1,2]})}}{P_{m^{[1,2]}(X^+_1 \in C^{[1]} | x^{[1,2]})} \text{.} \]

A more detailed description of this approach can be found in Zwitser and Maris (2015), and in Steinfeld and Robitzsch (2021a).

### CML Estimation in Probabilistic MST Designs

Based on the modification for deterministic routing outlined in the CML Estimation in Deterministic MST Designs section, the modification of the CML method in probabilistic MST designs (Steinfeld & Robitzsch, 2021b) can be described as follows. Let \( C^{[1]} \) be the event that the next module with score \( X^+_1 \) is chosen. As stated, instead of a deterministic cut score a probability vector \( p^{[b]}(x^{[b]}) \) is applied for the routing process. The probability \( P_{m^{[1,2]}(X^+_1 \in C^{[1]} | x^{[1,2]})} \), that person \( p \) with score \( x^+_1 \) in module \( m^{[1]} \) is routed to module \( m^{[2]} \) only with probability \( p^{[1]}(x^+_1) \) can be expressed as follows

\[
P_{m^{[1,2]}(X^+_1 \in C^{[1]} | x^{[1,2]})} = \frac{\sum^{j=0}_{j=0} P^{[1]}_j Y_j(m^{[1]} | y_{x^{[1,2]} - j}(m^{[2]})} \text{.} \]

In a probabilistic routing approach, however, there is no deterministic restriction on possible raw scores within each path of the MST design. Therefore, all raw scores with their respective probabilities – denoted in Equation 13 by \( p^{[1]}(j) \) – are considered in the calculation of the ESF.

Steinfeld & Robitzsch

Quantitative and Computational Methods in Behavioral Sciences
2023, Article e10087, https://doi.org/10.5964/qcmb.10087

Zwitser and Maris (2015)

Steinfeld and Robitzsch (2021a)
Implementation in R: The Package tmt

In the following, the package tmt will be introduced in detail. Here the modified CML approach as described in the CML Estimation in MST Designs section has been implemented. The package is developed for R (R Core Team, 2021), a language and environment for statistical computing, and a common software tool for psychometric and statistical analysis. The software R and the available packages are all published as open-source. Regarding psychometrics and in particular IRT, several R packages with a large variety of estimation methods for the estimation of item parameters are published (see, e.g., Baker & Kim, 2017; Bürkner, 2021; Chalmers, 2012; Choi & Asilkalkan, 2019; De Boeck et al., 2011; Fox, 2007; Hohensinn, 2018; Johnson, 2007; Mair & Hatzinger, 2007b; Paek & Cole, 2019; Rizopoulos, 2006).

As stated in the Introduction next to the CML parameter estimation method, the MML approach is often applied. Here several packages are available, a selection of those are, e.g., the package ltm (Rizopoulos, 2006) with the function ltm::rasch(), the package sirt (Robitzsch, 2021) with the function sirt::rasch.mml2(), the package TAM (Robitzsch et al., 2021) with the function TAM::tam.mml() or the package mirt (Chalmers, 2012) with the function mirt::mirt(). All these packages offer a variety of models which can be estimated. The function sirt::rasch.mirtlc() in the sirt package can be applied to estimate log-linear smoothing. Here the model type (e.g. modeltype = ‘MLCl’) and the trait distribution distribution.trait = ‘smooth4’ is passed as additional arguments.

This contribution focuses on the modified CML parameter estimation. For the common CML estimation, there are many R packages available like the well-known eRm Package with the main function eRm::RM() (Mair et al., 2021), the package psychotools with the function psychotools::raschmodel() (Zeileis et al., 2021), the package dexter with the function dexter::fit_enorm() (Maris et al., 2022), the package immer with the function immer::immer_cml() (Robitzsch & Steinfeld, 2018) and the package tmt with the function tmt::tmt_rm() (Steinfeld & Robitzsch, 2022), to name a few representatives. All packages have in common that they allow a user-friendly infrastructure but differ in speed and the availability of additional analysis options. Choi and Asilkalkan (2019) presented a comparative overview of some IRT packages (for application of different packages, see, e.g., Debelak et al., 2022).

Regarding the item parameter estimation with data obtained by an MST design, two R packages tmt (Steinfeld & Robitzsch, 2022) and dexterMST (Bechger et al., 2022) are currently available for deterministic routing utilizing the modified CML estimation method, while the probabilistic routing is currently only available in the package tmt. In the following, the main functions of the package tmt and its utilization are illustrated. The most recent version of the tmt package can be found in the Supplementary Materials.

First, the constructed model syntax for the specification of the MST design will be introduced. Second, the main package functions for the parameter estimation are outlined.
following some illustrations based on simulated data for different MST designs. A major motivation for the development process of tmt was keeping the parameter estimation as simple as possible for the user next to a large functionality of the package (the package is constantly being enhanced). Another aspect was the speed of the estimation process, even for larger amounts of items. For the first motivation, a model syntax was developed for ease of use, which will be presented below in detail. In terms of speed, the essential parts of the estimation process (essentially the calculation of the symmetric function introduced in the Parameter Estimation section were written in C++ utilizing the R package Rcpp (Eddelbuettel & Balamuta, 2018).

### MST Model Specification

As stated in the CML Estimation in MST Designs section, to apply the CML estimation method for MST designs, it is necessary to consider the restriction of the raw scores in each path. For this purpose, we developed a syntax to easily translate the applied MST design for data collection for the item parameter estimation. The language used in the model syntax is listed in Table 1. A short MST design to illustrate the translation from the MST design can be found in Listing 1.

<table>
<thead>
<tr>
<th>Formula Type</th>
<th>Syntax Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>module</td>
<td>=∼ m1 =~ c(i1, i2, i3, i4, i5) or m1 =~ paste0(‘i’,1:5)</td>
</tr>
<tr>
<td>path</td>
<td>:= pl := m2(min, max) or pl := m2(r1)</td>
</tr>
<tr>
<td>routing rule</td>
<td>= r1 = c(min, max) or c(probabilities)</td>
</tr>
<tr>
<td>sequential routing</td>
<td>+ pl := m2(min, max) + m1(min, max)</td>
</tr>
<tr>
<td>cumulative routing</td>
<td>++ pl := m2(min, max) ++ m1(min, max)</td>
</tr>
<tr>
<td>pre-condition</td>
<td>== data$variable</td>
</tr>
</tbody>
</table>

For the specification of modules, the syntax =∼ is used with the name of each module on the left-hand side (in Listing 1 indicated with ’m1’) and an R-vector with the containing items in the module on the right-hand side. As indicated in Table 1, there are several possibilities to specify the R-vector for convenience. Next, the paths and the applied routing rules must be specified. For the path (indicated here as ’p1’ and ’p2’), the syntax ‘:=’ is used. Here, the name of the respective path is put on the left-hand side, and the module constitutes the path on the right-hand side. Specifying the path also requires the specification of the routing rules and the type of routing (sequential or cumulative). For deterministic routing, the minimum and maximum raw scores per module must be indicated in parentheses after each module. The modules are then connected to a path.
with ‘+’ for sequential routing and ‘++’ in the event of cumulative routing (see again Table 1). The MST design used here is a simplified example for illustration. With the package tmt, it is also possible to estimate item parameters of more complex MST designs. Conceivable routing into more than one module, a path consists of several modules or routing from different paths into the same subsequent module. A slightly extended example of a more complex design with 40 items is shown in Listing 2. Here, in the second Stage three deterministic routing options to the third Stage are available.

**Listing 1**

*Example of the Used Model Syntax for an MST Design With Three Modules and Two Stages With Sequential Deterministic Routing*

```r
definition of the MST design in tmt:
# mstdesign <- 
# m1 <- c(i01,i02,i03,i04,i05)
# m2 <- c(i06,i07,i08,i09,i10)
# m3 <- c(i11,i12,i13,i14,i15)
# p1 := m2(0,2) + m1
# p2 := m2(3,5) + m3
```

**Listing 2**

*Example of a Slightly More Complex MST Design With Sequential Deterministic Routing*

```r
definition of the MST design in tmt:
# mstdesign <- 
# m4 <- paste0('i', 1:5)
# m2 <- paste0('i', 6:10)
# m5 <- paste0('i', 11:15)
# m1 <- paste0('i', 16:25)
# m6 <- paste0('i', 26:30)
# m3 <- paste0('i', 31:35)
# m7 <- paste0('i', 36:40)
# define path
# p1 := m1(0, 5) + m2(0, 1) + m4
# p2 := m1(0, 5) + m2(2, 3) + m5
# p3 := m1(0, 5) + m2(4, 5) + m6
# p4 := m1(6,10) + m3(0, 1) + m5
# p5 := m1(6,10) + m3(2, 3) + m6
# p6 := m1(6,10) + m3(4, 5) + m7
```

Estimating Item Parameters in MST with tmt 12

Quantitative and Computational Methods in Behavioral Sciences

2023, Article e10087, https://doi.org/10.5964/qcmb.10087
For sequential probabilistic MST designs, the deterministic routing rules must be replaced by the respective probabilities. These probabilities are specified for each possible raw score in the previous module, as illustrated in the following example in Listing 3, lines seven and eight (e.g. in ‘r1’, 0.9 is the probability applied for raw score 0; 0.76 for raw score 1; ...).

Listing 3

Example of the Used Model Syntax for an MST Design With Three Modules and Two Stages With Sequential Probabilistic Routing

```r
1 # definition of the MST design in tmt:
2 mstdesign ← "
3  m1 =~ c(i01,i02,i03,i04,i05)
4  m2 =~ c(i06,i07,i08,i09,i10)
5  m3 =~ c(i11,i12,i13,i14,i15)
6
7  r1 = c(0.9,0.76,0.62,0.48,0.34,0.2)
8  r2 = c(0.1,0.24,0.38,0.52,0.66,0.8)
9
10  p1 := m2(r1) + m1
11  p2 := m2(r2) + m3
12 "
```

Data Generation

The estimation function `tmt::tmt_rm()` expects either a $P \times I$ matrix with $P$ persons and $I$ items of the R-data-types matrix or data.frame. It is required here that the names of the columns follow the item names as specified in the respective multistage model. If required, it is also possible to add columns with additional information regarding the ability of persons, here referred to as pre-conditions. Some MST designs apply pre-tests, and questionnaires or incorporate other information for the routing process, which might be helpful for a valid selection of a suited routing module. This pre-information is only used for the routing but not for the parameter estimation. Together with the number-correct score of the routing module and the score of the pre-information a cumulative number-correct score is calculated, and additional modules are selected.

Parameter Estimation

In `tmt`, the item parameter estimation of data obtained by an MST design is straightforward. After the specification of the MST designs described in the MST Model Specification section, the data are prepared according to the description in the Data Generation section. Both the data and the translated MST design were handed over to the estimation function `tmt::tmt_rm()`. For the estimation, the unconstrained and box-constructed optimization using port routines is used (`nlminb` from the stats package in R; Fox et
al., 1978; Gay, 1990; R Core Team, 2021), as in our experience, this optimization seems to find the minimum while other optimization routines (here `stats::optim()`) does not. Singh and Dixit (2016) stated in their results that this algorithm is the method of choice for accuracy. They also suggest applying bounds for the parameters if available, to speed up the estimation process. The algorithm by Broyden-Fletcher-Goldfarb-Shanno (BFGS Fletcher, 1970) from the optimizer `optim` (from the `stats` package) can be alternatively applied. This is a quasi-Newton optimization method that approximates the Hessian matrix (by changing the value for the variable ‘optimization’ in the function `tmt::tmt_rm()` to `tmt::tmt_rm(optimization = 'optim')`).

Application of the `tmt` Package in a Nutshell

In the following, the application of the package `tmt` is illustrated for sequential and cumulative deterministic as well as probabilistic MST designs. For the demonstration, an MST with seven modules, four paths, and three stages is applied (see Figure 1). Each module contains ten items with different item difficulties. The routing module is module ‘m1’. First, an MST design with sequential deterministic routing is considered in the Illustration of Parameter Estimation in Sequential Deterministic MST Designs section, followed by a demonstration of cumulative deterministic routing in the Illustration of Parameter Estimation in Cumulative Deterministic MST Designs section. The same structure of the MST design used for the demonstration of deterministic routing is applied for probabilistic routing illustrated in the Illustration of Parameter Estimation in Sequential Probabilistic MST Designs and Illustration of Parameter Estimation in Cumulative Probabilistic MST Designs sections.

Illustration of Parameter Estimation in Sequential Deterministic MST Designs

For the demonstration of the package first both item (beta) and person (theta) parameters are generated. This step is shown in Listing 4.

Listing 4
Generating Item and Person Parameters for the Illustration

```r
1 library(tmt) # loading the package
2
3 # generate item parameters with corresponding names to the MST design above
4 beta ← seq(-2, 2, length.out = 70)
5 names(beta) ← paste0('i', seq_along(beta))
6
7 # generate person parameter
8 set.seed(6542) # the seed set only for illustration purposes
9 theta ← stats::rnorm(25000, 0, 1)
```

Quantitative and Computational Methods in Behavioral Sciences
2023, Article e10087, https://doi.org/10.5964/qcmb.10087
Considering the model syntax introduced in Table 1, the deterministic MST design with sequential deterministic routing is indicated in tmt by ‘+’. First, the modules are specified, followed by the paths built by modules. The respective cut score is specified in parentheses after each module in each path as illustrated in Listing 5.

**Listing 5**

*Specification of an MST Design With Sequential Deterministic Routing in tmt*

```r
1 # specification of MST design for tmt with deterministic sequential routing
2 mstdesign_m01 <- "
3 m4 =~ paste0('i',1:10)
4 m2 =~ paste0('i',11:20)
5 m5 =~ paste0('i',21:30)
6 m1 =~ paste0('i',31:40)
7 m6 =~ paste0('i',41:50)
8 m3 =~ paste0('i',51:60)
9 m7 =~ paste0('i',61:70)
10
11 # define path
12 p1 := m1(0, 5) + m2(0, 5) + m4
13 p2 := m1(0, 5) + m2(6,10) + m5
14 p3 := m1(6,10) + m3(0, 5) + m6
15 p4 := m1(6,10) + m3(6,10) + m7
16 "
```

In this example, the item parameters are generated in ascending difficulty (line 4 in Listing 4). The assignment of the items to the modules in Listing 5 is then defined in such a way that the entry module ‘m1’ contains difficulties in the middle range. The difficulties in the modules ‘m5’, ‘m2’, and ‘m4’ decrease, and in ‘m6’, ‘m3’, and ‘m7’ increase.

To generate data, the specific MST designs, item parameters, and person parameters are handed over to the function `tmt::tmt_sim()`. The argument ‘seed’ is only set for demonstration purposes as illustrated in Listing 6.

**Listing 6**

*Demonstration of the Simulation Function in tmt to Generate Data, Based on the Specified MST Design From Listing 5.*

```r
1 # generate data in tmt
2 dat_m01 <- tmt::tmt_sim(mstdesign = mstdesign_m01,
3   items = beta, persons = theta, seed = 6542)
```
For the item parameter estimation in \texttt{tmt}, the generated data are passed to the function \texttt{tmt::tmt\_rm()} . If requested some graphical illustrations can be applied for item inspections with \texttt{tmt::tmt\_gmc()} (see Listing 7). This type of plot is applied for (intuitive) differential item functioning (DIF; see also Holland & Wainer, 1993; Maris & Bechger, 2007; Millsap, 2011; Osterlind & Everson, 2009) inspection as proposed by Rasch (1960) to investigate measurement invariance (see also Debelak et al., 2022; Fischer & Molenaar, 1995; Mair & Hatzinger, 2007a, 2007b; Wright & Stone, 1999).^2

Figure 2

\textit{Two Illustrations of the Graphical Model Check for Intuitive Differential Item Functioning}

![Graphical model check](image)

Note. Graphical model check on the left (Figure 2a) is without default values, while that on the right (Figure 2b) is with the specification of additional options to emphasize items.

Comparing the estimation results above with the results of the package \texttt{dexterMST} shows that both packages lead to very close item parameter estimates (the underlying R scripts for the comparison can be found in the Supplementary Materials (Steinfeld & Robitzsch, 2023)). The mean absolute error (MAE) of the estimated item parameters was $\text{MAE} = 3.248 \times 10^{-5}$. Further simulations considering the parameter recovery can be found in Steinfeld and Robitzsch (2021a). Here, in addition to different MST designs, different ability distributions were considered. Across all conditions and sample sizes, the package \texttt{tmt} leads to asymptotically unbiased item parameter estimates.

^2) If the simulation function of \texttt{tmt} is applied, this step can be shortened by passing the generated object from the function \texttt{tmt::tmt\_sim()} to the function \texttt{tmt::tmt\_rm()}, in the example from Listings 7 it would be sufficient to write \texttt{m01\_tmt <- tmt::tmt\_rm(dat\_m01)}.


Illustration of Parameter Estimation in Cumulative Deterministic MST Designs

The syntax for cumulative deterministic routing is very similar to the sequential deterministic routing introduced in the Illustration of Parameter Estimation in Sequential Deterministic MST Designs section. Therefore, the description in the following is shortened.

Listing 7

Demonstration of Item Parameter Estimation in \texttt{tmt} (Only the results of the first six items are presented)

```r
1 # store the generated data
data_m01 ← dat_m01$data
data_m01$test
3
data_m01$test ← data_m01$data
data_m01$test
4
data_m01$test ← data_m01$data
data_m01$test
5
data_m01$test ← data_m01$data
data_m01$test
6
7 # estimate item parameter in \texttt{tmt}
8 m01_tmt ← tmt::tmt_rm(dat = data_m01, mstdesign = mstdesign_m01)
9
10 # results of the item parameter estimation
11 summary(m01_tmt)
12
13 # application of the Likelihood ratio test function
14 m01_tmt_lr ← tmt::tmt_lrtest(m01_tmt)
15
16 # plot results (see Figure 2a)
17 tmt::tmt_gmc(m01_tmt_lr)
18
19 # illustration of additional options for the plot, like emphasize of specific items
20 # with e.g. common item formats (see Figure 2b)
21 info ← rep(c('group_a','group_b'),each = 35)
22 names(info) ← paste0('i', seq_along(beta))
23 drop ← c('i1','i18','i20','i10') # option to drop items
24
drop ← c('i1','i18','i20','i10') # option to drop items
25
tmt::tmt_gmc(object = m01_tmt_lr,
ellipse = TRUE,
info = info,
don = drop,
title = 'graphical model check',
alpha = 0.05,
legendtitle = 'split criteria')
```

Illustration of Parameter Estimation in Cumulative Deterministic MST Designs

The syntax for cumulative deterministic routing is very similar to the sequential deterministic routing introduced in the Illustration of Parameter Estimation in Sequential Deterministic MST Designs section. Therefore, the description in the following is shortened.

Steinfeld & Robitzsch 17

Quantitative and Computational Methods in Behavioral Sciences
2023, Article e10087, https://doi.org/10.5964/qcmb.10087
to the parts that differ. For the following examples, the parameters generated in Listing 4 are used. For the specification of MST designs with cumulative deterministic routing, the syntax ‘++’ is used in tmt. After each module, the minimum and maximum cumulative raw score for the routing threshold is specified in parentheses, as illustrated in Listing 8.

Listing 8

*Specification of an MST Design With Cumulative Deterministic Routing*

```
1 # specification of MST design for tmt with cumulative deterministic routing
2 mstdesign_m02 <- "
3  m4 =~ paste0('i',1:10)
4  m2 =~ paste0('i',11:20)
5  m5 =~ paste0('i',21:30)
6  m1 =~ paste0('i',31:40)
7  m6 =~ paste0('i',41:50)
8  m3 =~ paste0('i',51:60)
9  m7 =~ paste0('i',61:70)
10
11  # define path
12  p1 := m1(0, 5) ++ m2(0,10) ++ m4
13  p2 := m1(0, 5) ++ m2(11,15) ++ m5
14  p3 := m1(6,10) ++ m3(6,15) ++ m6
15  p4 := m1(6,10) ++ m3(16,20) ++ m7
16 "
```

Data can again be generated using the available function `tmt::tmt_sim()` in the package tmt (see Listing 9).

Listing 9

*Demonstration of the Simulation Function in tmt to Generate Data Based on the Specified MST Design From Listing 8*

```
1 # generate data in tmt
2 dat_m02 <- tmt::tmt_sim(mstdesign = mstdesign_m02, 
3  items = beta, persons = theta, seed = 3657)
```
Listing 10

Demonstration of Item Parameter Estimation in \texttt{tmt} (Only the results of the first six items are presented)

```r
1 # store the generated data
data_m02 ← dat_m02$data
2
3 # estimate item parameter in tmt
5 m02_tmt ← tmt::tmt_rm(dat = data_m02, mstdesign = mstdesign_m02)
4
7 # results
8 summary(m02_tmt)
9 ## Call: tmt::tmt_rm(dat = dat_m02)
10
11 ## Results of Rasch model (mst) estimation:
12
13 ## Difficulty parameters:
14 ##               est.b_i1    est.b_i2    est.b_i3    est.b_i4    est.b_i5    est.b_i6
15 ## Estimate   -1.96365690 -1.91027776 -1.85010094 -1.82170568 -1.73675870 -1.65458688
16 ## Std. Error  0.02690224  0.02667968  0.02644528  0.02634067  0.02605033  0.02580121
```

The remaining parts of the item parameter estimation do not differ from that demonstrated in the Illustration of Parameter Estimation in Sequential Deterministic MST Designs section for sequential deterministic routing. The estimated item parameters from \texttt{tmt} and \texttt{dexterMST} are almost the same (see Listing 10). The MAE of the estimated item parameters was MAE = 0.0037.

Illustration of Parameter Estimation in Sequential Probabilistic MST Designs

The procedure for estimating item parameters in probabilistic MST designs is comparable to those of deterministic designs. For the demonstration, the generated parameters as illustrated in Listing 4 are used. As stated in the demonstration of deterministic MST designs, first the specific MST model must be specified, illustrated in Listing 11. Again, considering the model syntax introduced in Table 1, the only difference in the formulation of deterministic MST designs is that the probabilities for every achievable raw score must be specified as indicated in the following illustration (‘r1’ and ‘r2’), replacing the previously defined deterministic routing rules in each path.
Listing 11

*Specification of an MST Design With Sequential Probabilistic Routing*

```r
# specification of MST design for tmt with sequential probabilistic routing
mstdesign_m03 <- "
m4 =~ paste0('i',1:10)
m2 =~ paste0('i',11:20)
m5 =~ paste0('i',21:30)
m1 =~ paste0('i',31:40)
m6 =~ paste0('i',41:50)
m3 =~ paste0('i',51:60)
m7 =~ paste0('i',61:70)
"

# Specification of the probability for each raw score for the routing process. In
# this example persons with a raw score of 0 in module `m1' are routed to m2 with
# the probability 0.9 (r1) and with the probability of 0.1 to m2 (r2)
r1 = c(0.9,0.83,0.76,0.69,0.62,0.55,0.48,0.41,0.34,0.27,0.2)
r2 = c(0.1,0.17,0.24,0.31,0.38,0.45,0.52,0.6,0.73,0.8)

# definition of four paths
p1 := m1(r1) + m2(r1) + m4
p2 := m1(r1) + m2(r2) + m5
p3 := m1(r2) + m3(r1) + m6
p4 := m1(r2) + m3(r2) + m7
"
```

As before, data can be generated using the available function `tmt::tmt_sim()` (see Listing 12).

Listing 12

*Demonstration of the Simulation Function in tmt to Generate Data Based on the Specified MST Design From Listing 11*

```r
# generate data in tmt
# load Package tmt
library(tmt)

# generate item parameters with corresponding names to the MST design above
beta <- seq(-2, 2, length.out = 70)
names(beta) <- paste0('i', seq_along(beta))

# generate person parameter
set.seed(6542) # the seed sed only for illustration purposes
theta <- stats::rnorm(25000, 0, 1)

dat_m03 <- tmt::tmt_sim(mstdesign = mstdesign_m03,
                        items = beta, persons = theta, seed = 6542)
```
As illustrated in Listing 13, the item parameters can be estimated with the application of the function \texttt{tmt::tmt\_rm}().

**Listing 13**

*Demonstration of Item Parameter Estimation in \texttt{tmt} (Only the results of the first six items are presented)*

```r
1 # store the generated data
data_m03 <- dat_m03$data
2
3 # estimate item parameter in tmt
m03_tmt <- tmt::tmt_rm(dat = data_m03, mstdesign = mstdesign_m03)
4
5 # results of the item parameter estimation
summary(m03_tmt)
6
7 ## Call: tmt::tmt_rm(dat = data_m03, mstdesign = mstdesign_m03)
8
9 ## Results of Rasch model (mst) estimation:
10
11 ## Difficulty parameters:
12 ## est.b_i1  est.b_i2  est.b_i3  est.b_i4  est.b_i5  est.b_i6
13 ## Estimate  -2.08149306 -1.93868259 -1.89807931 -1.89807931 -1.78190613 -1.75924600
14 ## Std. Error  0.03390043  0.03294204  0.03269118  0.03269118  0.03202441  0.03190298
```

**Illustration of Parameter Estimation in Cumulative Probabilistic MST Designs**

As shown in the Illustration of Parameter Estimation in Cumulative Deterministic MST Designs section for deterministic routing, to indicate routing with cumulative scores, the operator ‘++’ is used for the specification of the paths in the MST design. As with sequential probabilistic routing, it is necessary to specify the probabilities for the routing with cumulative scores for each possible raw score. However, not only for each module as in the sequential case but for all possible raw scores, which can be reached with the actual and previous modules in each path (see Listing 14).

3) It is not necessary to pass the MST design (`mstdesign_m03`) as shown in Listing 13 if the data are generated with \texttt{tmt::tmt\_sim}, as the design is part of the returned object from that function.
Listing 14

Specification of an MST Design With Cumulative Probabilistic Routing

```r
1 # specification of MST design for tmt with cumulative probabilistic routing
2 mstdesign_m04 <- "
3  m4 =~ paste0('i',1:10)
4  m2 =~ paste0('i',11:20)
5  m5 =~ paste0('i',21:30)
6  m1 =~ paste0('i',31:40)
7  m6 =~ paste0('i',41:50)
8  m3 =~ paste0('i',51:60)
9  m7 =~ paste0('i',61:70)
10
11  # define routing criteria
12  r1 = c(0.9,0.83,0.76,0.69,0.62,0.55,0.48,0.41,0.34,0.27,0.2)
13  r2 = c(0.1,0.17,0.24,0.31,0.38,0.45,0.52,0.59,0.66,0.73,0.8)
14  r3 = c(0.9,0.83,0.76,0.69,0.62,0.55,0.48,0.41,0.34,0.27,0.2,0.9,0.83,0.76,0.69,0.62
15  ,0.55,0.48,0.41,0.34,0.27)
16  r4 = c(0.1,0.17,0.24,0.31,0.38,0.45,0.52,0.59,0.66,0.73,0.8,0.1,0.17,0.24,0.31,0.38
17  ,0.45,0.52,0.59,0.66,0.73)
18
19  # define path
20  p1 := m1(r1) ++ m2(r3) ++ m4
21  p2 := m1(r1) ++ m2(r4) ++ m5
22  p3 := m1(r2) ++ m3(r3) ++ m6
23  p4 := m1(r2) ++ m3(r4) ++ m7
24 "
```

As stated in the previous Sections, data can be generated using the available function for date generation `tmt::tmt_sim()` (see Listing 15).

Listing 15

Demonstration of the Simulation Function in `tmt` to Generate Data Based on the Specified MST Design From Listing 14

```r
1 # generate data in tmt
2 # load Package tmt
3 library(tmt)
4
5  # generate item parameters with corresponding names to the MST design above
6 beta <- seq(-2, 2, length.out = 70)
7 names(beta) <- paste0('i', seq_along(beta))
8
9  # generate person parameter
10 set.seed(6542) # the seed sed only for illustration purposes
11 theta <- stats::rnorm(25000, 0, 1)
12
13 dat_m04 <- tmt::tmt_sim(mstdesign = mstdesign_m04,
```
With the application of the function `tmt::tmt_rm()`, the item parameters can be estimated (see Listing 16).

**Listing 16**

*Demonstration of Item Parameter Estimation in tmt (Only the results of the first six items are presented)*

```r
1 # store the generated data
2 data_m04 ← dat_m04$data
3
4 # estimate item parameter in tmt
5 m04_tmt ← tmt::tmt_rm(dat = data_m04, mstdesign = mstdesign_m04)
6
7 # results
8 summary(m04_tmt)
9 ## Call: tmt::tmt_rm(dat = data_m04, mstdesign = mstdesign_m04)
10
11 ## Results of Rasch model (mst) estimation:
12
13 ## Difficulty parameters:
14 ##              est.b_i1    est.b_i2    est.b_i3    est.b_i4   est.b_i5    est.b_i6
15 ## Estimate   -1.9784103 -1.89534461 -1.84179454 -1.81459534 -1.7758122 -1.71712684
16 ## Std. Error  0.0322033  0.03156631  0.03117581  0.03098338  0.0307158  0.03032584
```

**Summary and Discussion**

This article introduces the application of the package `tmt` for item parameter estimation in MST designs. Together with `dexterMST`, `tmt` is an R package that implemented the modified CML estimation approach for deterministic MST designs (Zwitser & Maris, 2015). This modification is necessary to utilize the CML estimation method without obtaining severely biased item parameter estimates, as would be the case with the common CML estimation method in MST designs (Glas, 1988). While the first part of this article outlines the modification of the CML estimation, the second part illustrates the application and functionality of the package `tmt`. For the introduction of the estimation process, MST designs are simulated considering two different routing strategies to outline the model specification with the model syntax used in the package `tmt`.

Next to the deterministic routing approach, a separate section also discusses probabilistic routing strategies and their implementation in `tmt`. This strategy is applied, for example, in international educational large-scale assessments studies, to obtain e.g. a minimum number of item responses. For probabilistic routing again a modification of the CML method is necessary as proposed by Steinfeld and Robitzsch (2021b). Derived

Quantitative and Computational Methods in Behavioral Sciences.
2023, Article e10087, https://doi.org/10.5964/qcmb.10087
from the examples, the R package `tmt` provides asymptotically unbiased item parameter estimates in MST designs with deterministic and probabilistic routing strategies. Other examples can be found in the vignette and the supplemental material of the package `tmt` (Steinfeld & Robitzsch, 2022, 2023). As an outlook for future versions of the package `tmt`, extensions regarding usability are planned. Here, the automatic adaptation of the specified MST design should be highlighted. It is expected that missing values might occur especially in the last module, due to lack of time (not reached). In those cases, it is necessary to adapt the rules of the specified MST design for each occurring missing value pattern if those items should be kept as missing values and not recoded as not solved. This might be considered as a disadvantage compared to the MML method, which can be faced in a future version of the package `tmt` with an implemented algorithm for automatic adaptation of the MST design (see also Steinfeld & Robitzsch, 2021a, for a more detailed comparison of different estimation methods in MST designs). Furthermore, it is conceivable that in the next release, not only dichotomous but also polytomous scored items can be considered with the implementation of the partial credit model (Masters, 1982).

**Funding:** The authors have no funding to report.

**Acknowledgments:** The authors have no additional (i.e., non-financial) support to report.

**Competing Interests:** The authors declare no conflict of interest. The authors received no financial support for the research.

**Supplementary Materials**

The `tmt` R script vignettes are freely available at Steinfeld & Robitzsch (2023).

**Index of Supplementary Materials**

Steinfeld, J., & Robitzsch, A. (2023). *Supplementary materials to "Estimating item parameters in multistage designs with the tmt package in R" [tmt R script vignettes].* OSF. [https://doi.org/10.17605/OSF.IO/EZ87S](https://doi.org/10.17605/OSF.IO/EZ87S)

**References**


https://doi.org/10.1007/978-0-387-85461-8


Yamamoto, K., & Khorramdel, L. (2018). Introducing multistage adaptive testing into international large-scale assessments designs using the example of PIAAC. *Psychological Test and Assessment Modeling, 60*(3), 347–368.


