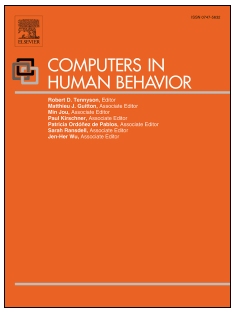
Accepted Manuscript



Stochastic programming for individualized test assembly with mixture response time models

Bernard P. Veldkamp, Marianna Avetisyan, Alexander Weissman, Jean-Paul Fox

PII: S0747-5632(17)30303-5

DOI: 10.1016/j.chb.2017.04.060

Reference: CHB 4959

To appear in: *Computers in Human Behavior*

Received Date: 30 June 2016

Revised Date: 24 April 2017

Accepted Date: 30 April 2017

Please cite this article as: Veldkamp B.P., Avetisyan M., Weissman A. & Fox J.-P., Stochastic programming for individualized test assembly with mixture response time models, *Computers in Human Behavior* (2017), doi: 10.1016/j.chb.2017.04.060.

Stochastic Programming for Individualized Test Assembly With Mixture Response Time Models

Bernard P Veldkamp[1]

Marianna Avetisyan[1]

Alexander Weissman[2]

Jean-Paul Fox[1]

[1] University of Twente, The Netherlands

[2] Law School Admission Council, Newtown, PA

Abstract

Early research on response time modeling assumed that a test taker would show consistent response time behavior, often referred to as working speed, over the course of a test. Such models may be unrealistic for various reasons — a warm-up effect may cause a test taker to respond more slowly than expected to the early items, fatigue may cause a test taker to respond more slowly than expected toward the end of a test, or as time runs out the test taker may quickly guess the answers to the last items on a test. To take these variations in working speed into account, mixture response time models have recently been investigated. Until now, mixture response time models have only been applied for post hoc analyses. This research expands the use of these models by exploring their application in the context of the assembly of individualized computer-based assessments (CBAs). Response time constraints are probabilistic in nature. Stochastic programming was compared to three existing strategies for dealing with probabilistic constraints. Stochastic programming proved to be a very suitable strategy for solving test assembly problems with mixture response time models. Using stochastic programming, computer-based tests could be assembled in such a way that response time information could be used to the fullest extent.

Introduction

With computerized test administration becoming increasingly popular in educational measurement, more detailed information regarding the response behavior of test takers is becoming available. Along with recording the endorsed or selected item responses, test administration log files can also provide information about test takers' response times, how they select or eliminate potential response options, the order in which they answer items, and how they utilize auxiliary materials. Even more detailed information, such as individual mouse clicks and key strokes, is available for analysis. Nevertheless, several difficulties must be overcome in retrieving information from log file data (Greiff, Niepel, Scherer, & Martin, 2016; OECD, Chap. 7, 2015). Further, it remains a challenging task to analyze and interpret the information (He, & von Davier, 2014; Timmers, Walraven, & Veldkamp, 2015).

Compared to other sources of information, analyzing response times (RTs) is relatively straightforward to conduct. When only one item is administered at a time, log files provide accurate response time information at the item level. In fact, there has been quite a tradition in RT modeling (e.g., Hornke, 1997, Masters & Keeves, 1999, Schnipke & Scrams, 1997). RTs can provide information about the average speed of working, the speededness of a test toward the end, warm-up effects, and fatigue (Ackerman & Kanfer, 2009; Evans & Reilly, 1973; Lawrence, 1993; van der Linden, 2011). For example, long RTs at the beginning of a test, often in combination with relatively many mistakes, may be an indication of a warm-up effect. Short RTs toward the end of a test, often in combination with a high number of mistakes, are an indication that the test might be speeded. Moreover, long RTs and relatively many mistakes toward the end

of the test might indicate fatigue. Recently, Lee and Jia (2014) applied RTs to analyze test-taking behavior in large-scale assessments.

RTs can also be used for item selection. Fan, Wang, Chang, and Douglas (2012) introduced information per time unit as a new index for item selection in computerized adaptive testing (CAT). When the total RT for a test is restricted, selecting items based on maximum Fisher information might not be the most efficient approach. Imagine an item bank where the most informative item provides 5% more information than any other item but is three times more time consuming to answer. In that case, it might be more economical to administer a larger number of items with shorter RTs, which taken together provide more information than the most informative item. RTs also reveal which items are more sensitive to working speed than others (Marianti, Fox, Avetisyan, & Veldkamp, 2014). Taking into account the sensitivity of items to working speed can very useful in the test development process; for example, in helping to prevent situations where some test takers might run out of time while others are able to demonstrate their ability and finish the test without time pressure. Recently, Finkelman, Kim, Weissman & Cook (2014) published a paper on item selection for cognitive diagnostic models and CAT in which RTs were taken into account.

Finally, RTs have been used to identify aberrant response behavior such as cheating (van der Linden & van Krimpen-Stoop, 2003). Very short RTs combined with correct answers to relatively difficulty items by low or average ability test takers, for example, are generally seen as a strong indication of cheating. Van der Linden and Guo (2008) mention three reasons why RTs are a strong source of information about aberrant response behavior: (a) they are very suitable for statistical testing because they are continuous variables; (b) in CAT, it remains possible to distinguish likely from unlikely RT patterns; and (c) even when test takers try to simulate

realistic RTs, it is almost impossible for them to find out what a typical RT pattern would be at their ability level. Several checks (Levine & Rubin, 1979; van der Linden & Guo, 2008) have been proposed to test for aberrant RTs.

Various RT models have been presented in the literature. Some RT models have been presented that only focus on the RTs, without taking the correctness of the response into account (e.g., Maris, 1993; Schnipke & Scrams, 1997). Another category of models focuses on both RTs and accuracy. Van der Linden (2006, 2007) introduced a hierarchical framework for modeling both speed and accuracy concurrently. In this framework, a normal ogive model is formulated for dealing with the responses, a lognormal model is chosen for the RTs, and a bivariate normal distribution is chosen to model the joint distribution of both person and item parameters.

One practical concern is how to apply RT models to operational computer-based assessments, particularly for individualized test assembly such as multi-stage testing, computerized adaptive testing, or when individual linear test forms are assembled on-the-fly for every candidate, without knowing a candidate's proficiency in advance. Fan et al. (2012), van der Linden, Scrams, and Schnipke (1999), van der Linden (2011), and Veldkamp (2016) proposed different models to take RTs into account during test assembly. They illustrated how RTs could be used to adapt item selection to the working speed of the candidates in order to avoid speededness issues towards the end of the test. All these papers assume a uniform working speed for the candidates. More recently, mixture or dynamic RT models have been introduced to relax the requirement of a uniform working speed (Marianti et al., 2014; Molenaar & de Boeck, 2014; Fox, 2014), and while the development of such models is still in its infancy, the models fit the data well and have enabled researchers to interpret the observed response behavior.

Until now, mixture RT models have only been applied for post hoc analyses. A reasonable next step would be to use them during test development or during test administration. For example, when pretesting or prior experience has indicated that a significant number of test takers have shown differential speed behavior due to warm-up effects, speededness toward the end of the test, or fatigue, such factors might be taken into account in test development. Another example relates to CAT, where the application of mixture RT models might reveal aberrant test-taker behavior. If cheating is suspected, immediate actions might be taken for the remaining administered items, for example, by selecting items from a secret back-up pool of previously non-administered items that have been put aside for such situations.

One way to apply mixture RT models to test development would be to calibrate items with a mixture RT model beforehand. When the items are selected from a calibrated item bank, the RTs can be analyzed during testing. Such a procedure would be comparable to a regular multi-stage test or a computerized adaptive test, where a test taker's ability is estimated during testing, and the estimated ability is used to make decisions about administering the subsequent items. Application of mixture RT models in the assembly of individualized computer-based assessments (CBAs) would enable a testing procedure that uses response times information to the fullest extent.

Incorporating mixture RT models in test assembly introduces some complexity. Mixture RT models can't be used in the assembly of CBAs straightforwardly, since the models distinguish several latent classes of response times behavior and it is unknown in advance to which class the response behavior of a specific test taker belongs or how the response behavior of a group of test takers is distributed over these classes. This paper studies various strategies to assemble CBAs in the case of mixture RT models. Stochastic programming is introduced as a new strategy for

dealing with mixture RT models in CBA assembly and the performance of stochastic programming is compared to the existing strategies in a simulation study.

First, an overview of response time models is provided. Following this overview is a discussion of mixture models, which then leads into the presentation of mixture response time models. A model for the assembly of CBAs in the case of a mixture RT model is presented next. Three existing strategies for solving the test assembly model are described, and stochastic programming is introduced for solving the test assembly models with uncertainty in the constraints. In a simulation study based on an operational CBA, application of the strategies is illustrated and evaluated. Recommendations about its use are given. Finally, a generalization of these techniques to more general test assembly problems is discussed.

Response time models

Response times typically have a positive skewed distribution, with response times (RTs) truncated at zero, since negative response times are not possible. Most test takers' RTs will be located in the vicinity of the mode, and a minority of the test takers might have much higher response times. To model this distribution effectively, a lognormal response time model is applied. This means that the logarithm of the response times is assumed to be normally distributed. In the lognormal model, the RT distribution can be characterized by the working speed parameter ς_p, a time-intensity parameter λ_i, and a time-discrimination parameter ϕ_i, where *p* indexes persons and *i* indexes items. For a given person *p*, the working speed is assumed to be constant during the test. The time-intensity parameter is a measure of the time needed to complete the item, and the time-discrimination parameter is a measure of the

sensitivity of the item to differences in working speed between test takers. Since the response behavior might vary due to distraction, fatigue, or other causes, and because these deviations are assumed to be independent of working speed, a normally distributed measurement error component is added to the model, with mean equal to zero and variance equal to σ_i^2. When the observed RTs of person p to item i are denoted by T_{ip}, the lognormal RT model can be formulated as

$$p(t_{ip} \mid \varsigma_p, \phi_i, \lambda_i, \sigma_i^2) = \frac{1}{t_{ip}\sqrt{2\pi\sigma_i^2}} \exp\left[-\frac{1}{2\sigma_i^2} \left(\ln t_{ip} - \phi_i(\lambda_i - \varsigma_p) \right)^2 \right] \quad (1)$$

This formulation of the lognormal RT model deviates slightly from the model in van der Linden (2006), since a time-discrimination parameter is added. Besides, this model parameterizes the time-discrimination slightly differently from Van der Linden (2009), who introduced $1/\sigma_i$ as time discrimination (Marianti et al., 2014).

Van der Linden (2007) proposed a hierarchical framework for modeling both the probability of response correctness and RTs concurrently. In this framework, a normal ogive model is applied for modeling correctness and a lognormal model for RTs. To model the joint distribution of both person and item parameters, a bivariate normal distribution is assumed. Bayesian estimation procedures can be applied to estimate the models. Technical details about the Markov Chain Monte Carlo (MCMC) methods, the specifications of prior distributions for the parameters, and the full conditional distributions of the model parameters can be found in Fox, Klein Entink, and van der Linden (2007), Klein Entink, Fox, and van der Linden (2009), and van der Linden (2006, 2007).

One of the assumptions in van der Linden's model is that test takers work at uniform speed during test administration (van der Linden, 2009). In practice this assumption might not hold, since test takers might need to warm up or they might get tired during test administration. Marianti et al. (2014), Molenaar and de Boeck (2014), and Fox (2014) proposed using a mixture of response models or a more dynamic RT model to describe the RT behavior of the test takers. They assumed that the working speed of a test taker varies during the test, and that it is related to, for example, test-taking strategy or varying circumstances. Marianti et al. (2014) presented an example where some test takers showed aberrant response behavior. Molenaar and de Boeck (2014) studied the case where test takers used different more or less efficient strategies for solving the items, alternating among various strategies during the test (see Partchev & De Boeck, 2012). Finally, Fox (2014) introduced a dynamic model that accounted for the behavior where test takers increased or decreased their working speed during test administration. The next section discusses mixture models more generally, followed by a section discussing mixture RT models.

Mixture models

Mixture models have been applied in both educational and psychological measurement to account for different response behavior by various groups of respondents in the population (Hancock & Samuelsen, 2008). These groups are also referred to as latent classes, since it cannot be observed directly to which class a respondent belongs. Statistical methods have to be applied to identify these classes, where respondents in one latent class behave more alike than respondents from different classes. Several examples can be found in the literature. Mixture IRT

models have been applied for explanatory DIF analysis by Cohen & Bolt (2005) and by Cho & Cohen (2010). Egberink, Meijer, & Veldkamp (2010) applied mixture IRT modeling to a Conscientiousness scale and found that this construct is qualitatively different for different groups of respondents, which influenced their response style. The software packages Mplus (Muthén & Muthén, 2012) and Latent Gold (Vermunt & Lagidson, 2013) are generally applied for the analysis of mixture models.

Mixture response time models

In mixture RT models, a multicomponent distribution can be defined to account for the differences in the RT behavior of various classes of test takers. Several examples of mixture RT modeling can be found in the literature.

Between-Subjects Latent Class Model

Marianti et al. (2014) describe a distinction between a class of test takers who behave according to the RT model and a class of test takers with aberrant behavior. For this mixture RT model, it holds that the response behavior of the test takers can be classified into a number of latent classes $A_0, \ldots, A_K$, where the classes A_k, $k = 0,\ldots,K$ are mutually exclusive and exhaustive. The probability of a person p's membership in a latent class A_k is known and denoted as $P(p \in A_k)$, a person can only be a member of one class, and the actual membership is a priori unknown.

According to the specifications of the mixture distribution, the RTs can be modeled as

$$P\left(T_{ip} = t_{ip} \middle| \zeta_p, \lambda_i, \phi_i, \sigma_i^2\right) = \sum_k P\left(T_{ip} = t_{ip} \middle| \zeta_p^{(k)}, \lambda_i^{(k)}, \phi_i^{(k)}, \sigma_i^{(k)2}, A_k\right) P(p \in A_k), \qquad (2)$$

where for each class k a lognormal RT model (see Equation 1) is estimated, and $\zeta_p^{(k)}$, $\lambda_i^{(k)}$, $\phi_i^{(k)}$, and $\sigma_i^{(k)}$ are the respective person and item parameters for the RT model of latent class *k*. It is even possible that different model formulations can be used for the various classes. For example, in Marianti et al (2014), a lognormal RT model is used to describe the behavior of latent class A_0, because that class represents regular behavior, and the RT model for latent class A_1 is a generic probability model, since it describes all possible aberrant RT behaviors.

Within-Subjects Latent Class Model

Molenaar and De Boeck (2014) chose a rather different approach. First, they assumed that the lognormal RT model in (1) can be used to describe the RTs. Then, they distinguished between fast and slow response behavior, where a test taker is allowed to alternate between fast and slow response behavior depending on whether the test taker behaves according to his or her fast ability $\theta_p^{(f)}$ or slow ability $\theta_p^{(s)}$. In contrast, in Marianti et al. (2014), test takers can only be a member of one class, each individual response is assigned to one of two classes, and class membership is a priori unknown. For each of the abilities in the approach of Molenaar and De Boeck, a separate measurement model can be defined:

$$\text{logit}\left[P_i(X_{ip} = 1 | \theta_p^{(s)}\right] = a_i^{(s)}\left(\theta_p^{(s)} - b_i^{(s)}\right), \qquad (3)$$

and

$$\text{logit}\left[P_i(X_{ip}=1|\theta_p^{(f)}\right]=a_i^{(f)}\left(\theta_p^{(f)}-b_i^{(f)}\right), \quad (4)$$

where $a_i^{(s)}$ and $a_i^{(f)}$ denote the respective discrimination parameters, and $b_i^{(s)}$ and $b_i^{(f)}$ denote the respective difficulty parameters. The probability of a correct response can now be modeled as

$$P\left(X_{ip}=1|\theta_p^{(s)},\theta_p^{(f)}\right)=\pi_{ip}P\left(X_{ip}=1|\theta_p^{(s)}\right)+(1-\pi_{ip})P\left(X_{ip}=1|\theta_p^{(f)}\right), \quad (5)$$

where π_{ip} denotes the probability that the test taker p answers item i using slow response behavior. In this model, the probabilities π_{ip} are made dependent on the RTs, while accounting for the main effects of items and persons on the RT distribution. Using the Block Design subtest from the Wechsler Intelligence Scale for Children IV (WISC-IV; Wechsler, 2003), and after imposing some restrictions on the parameters to prevent identification problems, Molenaar and De Boeck (2014) were able to explain the observed differences between the fast and slow responses of individual test takers.

Dynamic Factor Model

The Dynamic Factor Model for stochastic speed processes is described in Fox (2014). In this model, a test is assumed to consist of a number of blocks of items, each block having its own

average block working speed. Items in a block can be consecutive or spread out over the test. Depending on the underlying correlation between block speeds, they define a specific speed trajectory over blocks. In this way, RT models can account for variable working speed processes. For example, test takers may increase their working speed during a test when they run out of time toward the end of the test, or they may decrease it and work more slowly toward the end of the test due to fatigue. The main advantage of the Dynamic Factor Model is its flexibility in dealing with different kinds of nonstationary or stationary RT behavior of the test takers.

Implications of Mixture RT Modeling

In each of the examples above, researchers proposed using a more flexible RT model to investigate potential heterogeneity in the working speed of the test takers. The mixture RT models allowed for the investigation of groups of test takers who showed different speed behavior over the course of a test. Application of mixture RT models to real datasets has not only led to more precise measurement, but also enabled researchers to interpret the observed RTs.

Test Assembly

This section discusses how to assemble tests when mixture RT models have been used to calibrate an item bank. First a general model for test assembly is presented. The modifications that are needed to apply mixture RT models are then described and the implications for automated test assembly discussed.

In automated test assembly (ATA), 0-1 linear programming methods are generally applied for item selection. Van der Linden (2005) presented a general 0-1 LP model for the assembly of linear test forms. In this model, test specifications are modeled as constraints, and the objective function represents one of the attributes of the test that must be optimized in test development. The constraints of the model can be categorized as categorical, quantitative, or logical. Categorical constraints concern attributes of the items that categorize the item bank, such as content classification of the items or item type. Quantitative constraints concern attributes that have quantitative values, such as word count; RT constraints fall under this class of specifications. Finally, logical constraints deal with dependences between pairs or groups of items, such as sets containing enemy pairs, where one item provides clues for solving the other item, or item sets where multiple items are related to the same stimulus. The most common objective functions are maximization of test information, minimization of the deviation between the test information function and a pre-specified target, and minimization of the number of items.

Let $i = 1,...,I$ be an index for the items in the bank, x_i represent whether item *i* is selected or not, $I_i(\theta_p)$ be the amount of information provided by item i for person p with ability level θ_p, $c = 1,...,C$ be an index for the categories, bc_c be the maximum number of items that can be selected for category c, q_i denote the contribution of item i to quantitative attribute $q = 1,...,Q$, bq_q be the upper bound for attribute q, $e = 1,...,E$ be an index for the various logical constraints, and be_e be the maximum number of items to be selected for this group. For example, for an enemy constraint about two or more items that contain clues about each other such that only one of them can be selected for a specific test, this number is equal to one; for item sets, it is equal to the maximum number of items that can be selected from an item set. The model can now be formulated as:

$$\max \sum_{i=1}^{I} I_i(\theta_p) x_i \tag{6}$$

subject to

$$\sum_{i \in C} x_i \leq bc_c \quad c = 1, .., C, \tag{7}$$

$$\sum_{i=1}^{I} q_i x_i \leq bq_q, \quad q = 1, .., Q, \tag{8}$$

$$\sum_{i \in S_e} x_i \leq be_e, \quad e = 1, .., E, \tag{9}$$

$$x_i \in \{0,1\}. \tag{10}$$

This can be seen as a general formulation of a test assembly model, since any minimization of the objective function can be reformulated as a maximizing the negative of the original objective function. Besides, any lower bound can be reformulated as an upper bound by adding negative signs to both sides of the constraints. Finally, equality constraints can also be formulated as upper bound constraints, since '$=$' implies that both '$\leq$' and '$\geq$' hold. This general test assembly model can easily be modified and extended to be applicable for the assembly of multistage tests, computerized adaptive tests with constraints, tests measuring multiple traits, mastery tests, or even test batteries. For an overview of test assembly models, see van der Linden (2005).

Response time model constraints

Specifications related to RTs can be formulated to ensure that test takers can finish the test within the allotted time. Since working speed varies over candidates and some candidates have a tendency to postpone responding to a question and to wait for some insight to occur when they don't know the correct answer, most testing agencies apply specifications such as (a) 100% of the test takers must be able to respond to 90% of the items, or (b) 85% of the test takers must be able to respond to all of the items. In addition, testing agencies might want to help prevent situations where some test takers might run out of time while others are able to demonstrate their ability and finish the test without time pressure This might be an issue, for example, in CAT or multistage testing, where more capable test takers have to respond to more difficult, and often more time-intensive, items (van der Linden, 2006).

In automated test assembly, RT constraints are modeled in terms of expected RTs. The exact RT of a test taker is unknown beforehand, but the expected RT can be calculated using the modeled distribution for a test taker's RTs. For the lognormal RT model, using the parameterization in (1), the expected RT for item i of a test taker with working speed ς_p is equal to

$$E\left(t_{ip}\middle|\zeta_P\right) = \exp \phi_i(\lambda_i - \zeta_p + \frac{\sigma_i^2}{2}) \tag{11}$$

Let $E[T_{ip}]$ denote the expected RT for item i and test taker p, and let $T_{\max}$ be an upper bound for the available time. A generic formulation for RT constraints would be:

$$\sum_{i=1}^{I} E[T_{ip}]x_i \leq T_{max}. \tag{12}$$

In this formulation, the RT constraint holds for test takers *p*. By varying either the percentage of test takers or the percentage of items for which this constraint holds, this generic constraint can be applied to model most of the RT specifications encountered in practice.

The assembly of CATs can be seen as solving a series of linear test assembly problems when the shadow test approach (van der linden & Reese, 1998) is applied. For formulating generic RT constraints in CAT or multistage testing, see Appendix A.

Mixture response time model constraints

When mixture RT models are applied, formulation of RT constraints becomes slightly more complicated. Instead of one lognormal RT model that holds for all test takers, a mixture of models must be taken into account. The expected RT $E[T_{ip}]$ can now be calculated as:

$$E\left(t_{ip}\middle|\varsigma_p^{(1)},\ldots,\varsigma_p^{(K)}\right)=\sum_k \exp\phi_i^{(k)}\left(\lambda_i^{(k)}-\zeta_p^{(k)}+\frac{\sigma_i^{(k)2}}{2}\right)P(p\in A_k). \quad (13)$$

This implies that, instead of a single RT constraint, a mixture of RT constraints is defined:

$$\sum_{i=1}^{I}\left[\sum_k \exp\phi_i^{(k)}\left(\lambda_i^{(k)}-\zeta_p^{(k)}+\frac{\sigma_i^{(k)2}}{2}\right)P(p\in A_k)\right]x_i\le T_{max}. \quad (14)$$

Unfortunately, these probabilistic constraints cannot be handled by regular 0-1 LP methods directly, since these methods have been developed to deal with deterministic objective functions and constraints, which are linear functions of the decision variables x_i. As a consequence, alternative test assembly methods must be applied.

Several strategies for dealing with probabilistic optimization problems are available (Birge & Louveaux, 1997). First of all, a probabilistic mixture RT constraint can be reformulated into a deterministic one by using the average expected RT over all classes. The resulting constraint can now be formulated as:

$$\sum_{i=1}^{I} ave(E[T_{ip}])x_i \leq T_{max}. \qquad (15)$$

Where $ave(E[T_{ip}])$ denotes the average expected RT for person p to item i over all RT classes A_k. A drawback of this strategy is that a violation of probabilistic constraints is accepted for part of the population. To prevent these violations, a much more conservative reformulation of the model can be applied. The probabilistic constraint can be replaced by a series of deterministic constraints:

$$\sum_{i=1}^{I} \exp \phi_i^{(k)}(\lambda_i^{(k)} - \zeta_p^{(k)} + \frac{\sigma_i^{(k)2}}{2})x_i \leq T_{max}, \qquad \forall k. \qquad (16)$$

Unfortunately, one needs k times as many constraints in this approach. Besides, since the RT classes don't overlap, this strategy would severely over-constrain the problem. Finally, Bertsimas and Sim (2003) proposed robust optimization. They argued that maximum uncertainty only

impacts a final solution of any optimization problem for a limited number of items. In this method, a model with uncertainty in it is reformulated into a series of deterministic optimization problems.

Veldkamp (2013) described the application of robust optimization to automated test assembly problems. In robust optimization, the average expected RTs $\overline{E[T_{ip}]}$, the maximum expected RT over all classes $E[T_{ip}]^{\max}$, and the differences d_i between them must be calculated first for each item. Then, the items must be ranked based on their contribution to the objective function (6). Let Γ be the number of items for which uncertainty affects the solution. For most test assembly problems Γ can be set equal to 40% of the test length. Next, a series of l optimization problems, where $l = 1, \ldots,$ test length must be solved. In these problems, the following RT constraint is imposed:

$$\sum_{i=1}^{l} ave(E[T_{ip}])x_i + [\sum_{i=1}^{l}(d_i - d_i^*)x_i + \Gamma d_l^*] \leq T_{max} \qquad (17)$$

where $d_l^* = \min_{i<l}\{d_i\}$. Finally, the best solution of these l problems is chosen. Even though the uncertainties in the model are taken into account during optimization, a solution that is too conservative is prevented. The only drawback of this strategy is that a series of l optimization problems must be solved instead of one. For a detailed description and an analysis of the performance, the reader is referred to Bertsimas and Sim (2003).

What all of these strategies have in common is that they reformulate the model such that standard 0-1 LP software can be applied for solving the problem. However, reformulating the model comes at a cost. The final solution either violates the constraints, is far too conservative,

or is far more time consuming to obtain. In the next section, stochastic programming is introduced for dealing with mixture RT constraints.

Stochastic Programming

Stochastic linear programming deals with problems with random constraints (Klein Haneveld & van der Vlerk, 2006):

$$\max \sum_{i=1}^{I} c_i x_i \tag{18}$$

such that

$$\sum_{i=1}^{I} T(\omega)_i x_i \le h(\omega), \quad \forall \omega \in \Omega, \tag{19}$$

$$\sum_{i=1}^{I} a_{ij} x_i \le b_j, \tag{20}$$

$$x_i \in \{0,1\}, \quad i = 1,...,I, \tag{21}$$

where the actual value of ω (where ω might refer to, for example, the RT class to which the test taker belongs), is unknown. Only probabilistic information about ω is available; that is, we assume that the distribution of ω is given, and that Ω represents the set of all possible values. This model must be interpreted in the following way. In the first stage, we must decide on the first-stage variables x_i, without any information about the realization of ω available. However,

this solution will often be infeasible with respect to the second-stage specifications in (19) (Klein Haneveld & van der Vlerk, 1999). In our simulation study, settings from an operational computerized high-stakes test are used, where for each test taker a new linear test form is assembled from an item bank. These test forms have to meet a set of test specifications. For this test, a mixture RT model as described in (2) can be applied, and the total testing time is restricted. When the test is assembled, the RT classes to which the test taker belongs is unknown. So although ω is unknown, we do have a distribution of class membership for the whole population of test takers. In the first stage, we have to decide on variables x_i, denoting whether item i is selected for the test or not, without any information about the RT class membership of the individual test taker. *Note*: Since the total test time is identical for all test takers, ω only plays a role in the left-hand side of (19) for the example in the simulation study.

Two classical strategies for dealing with stochastic programming problems are available:

1. Penalty costs are assigned to violations of the constraints in (19). In this strategy, the objective function of the model in (18) is extended with a penalty function consisting of an expected violation of the probabilistic constraint multiplied by a cost parameter. Recourse actions are then taken to compensate for the infeasibility. Therefore, such strategies are also referred to as *recourse models* (Birge & Louveaux, 1997).
2. The second strategy is to specify a model with *chance constraints* (see also Birge & Louveaux, 1997). In these constraints, the probability that any of the constraints in (19) is violated is restricted:

$$P\left(\sum_{i=1}^{I} T(\omega)_i x_i > h(\omega)\right) \leq \alpha, \tag{22}$$

where α limits the probability of a violation. Imposing these constraints can be interpreted as accepting a solution to the problem in (18) through (21) as long as the solution is not too risky.

The second strategy seems more appropriate to apply to mixture RT modeling. As was also mentioned in the introduction section, RT specifications are often formulated relative to the population. Generally, the maximum amount of time to finish a test is limited for practical reasons. Within this time limit, a pre-specified percentage of test takers must be able to complete the whole test. At the individual level, this means that the probability of not finishing the test in time must be limited, which is exactly what chance constraints intend to model. When this strategy is applied in the context of test assembly with a mixture response times model, $T(\omega)_i$ indicates the average response time for a person in class ω to item i, and $h(\omega)$ refers to the time limit for a person in class ω. In case of identical time limits for all classes, it holds that $h(\omega) = T_{max}$.

The introduction of chance constraints to an optimization model introduces some technical difficulties. Chance constraints are nonconvex in general, especially when ω is discrete in nature, which is the case in mixture RT modeling where ω refers to class membership, and classes are assumed to be mutually exclusive and exhaustive. Therefore, Klein Haneveld and van der Vlerk (2002) proposed modeling them as *integrated chance constraints*, where the uncertainty in ω is integrated out of the constraint*:*

$$E\left[\sum_{i=1}^{I} T(\omega)_i x_i - h(\omega) \geq 0\right] \leq \beta, \tag{23}$$

where β represents the largest acceptable expected violation, and it is specified a priori by the test developer. Given that RT distributions are known for the various classes, values for β can be derived. For cases where the RT distribution for one of the classes is unspecified (Marianti et al., 2014), a decision must be made as to whether and how a bound should be imposed.

The integrated chance constraints strategy to stochastic programming has a similar functional form as the average expected RT constraints strategy in (15) for dealing with the mixture of RT constraints in (14). When the largest acceptable expected violation is set equal to $\beta = 0$, they are even identical. It should be noted however, that both strategies come from entirely different perspectives. The average expected RT constraints strategy in (15) is formulated in such a way that the average candidate in a population is expected to meet the time limits, whereas the integrated chance constraint allows possible violation of the time limit at the individual level, since it is unknown to which RT class the candidate belongs.

Simulation Study

Two simulation studies were carried out to illustrate the use of stochastic programming, and to test whether and when stochastic programming would be beneficial in automated test assembly. Real items from a Basic Safety Exam (BSE) were used to generate the item bank. This exam consists of 40 knowledge items, and it is obligatory for all personnel of petrochemical factories in the Netherlands. Annually, thousands of test takers participate in this exam. The exam is

administered both on paper and digitally, so detailed information about actual RTs is available. In the digital environment, an individual linear test form is assembled for each candidate using stratified random sampling from an item bank of 1,700 items to prevent cheating. This means that in BSE administration, individual linear test forms are assembled on-the-fly for every candidate. These test forms are non-adaptive, since they are assembled prior to administration and hence no information about examinee performance is available. Independence between ability and speed is assumed. The R package LNIRT (Marianti, 2015) was used to estimate the RT model parameters. The MIRT package (Glas, 2010) was used to calibrate the IRT parameters using a two-parameter logistic model. We were not allowed to publish the real item parameters for this test, but based on the real item parameter estimates and their distributions, we simulated a pool of 640 items. The test is administered in over ten different languages, and each test taker can take the test in his/her mother tongue. The item parameters of these versions are more or less comparable (in the exam they are assumed to be equal for all languages), but some of the languages are more time-intensive than others. Therefore we simulated parameters for a mixture RT model where we distinguished between a fast (85% of the population) and a slow (15% of the population) working speed. We used the same item parameters for both classes, $a \in [0.29, 1.12]$ and $b \in [-2.60, -0.48]$; and a different RT parameter for fast response behavior, $\phi_f \in [0.66, 1.42]$ and $\lambda_f \in [3.02, 3.69]$, and for slow response behavior, $\phi_s \in [0.67, 1.41]$ and $\lambda_s \in [3.4, 3.91]$. For a distribution of both the item and the RT parameters, see Figure 1. Note that both simulation studies employed the same item bank.

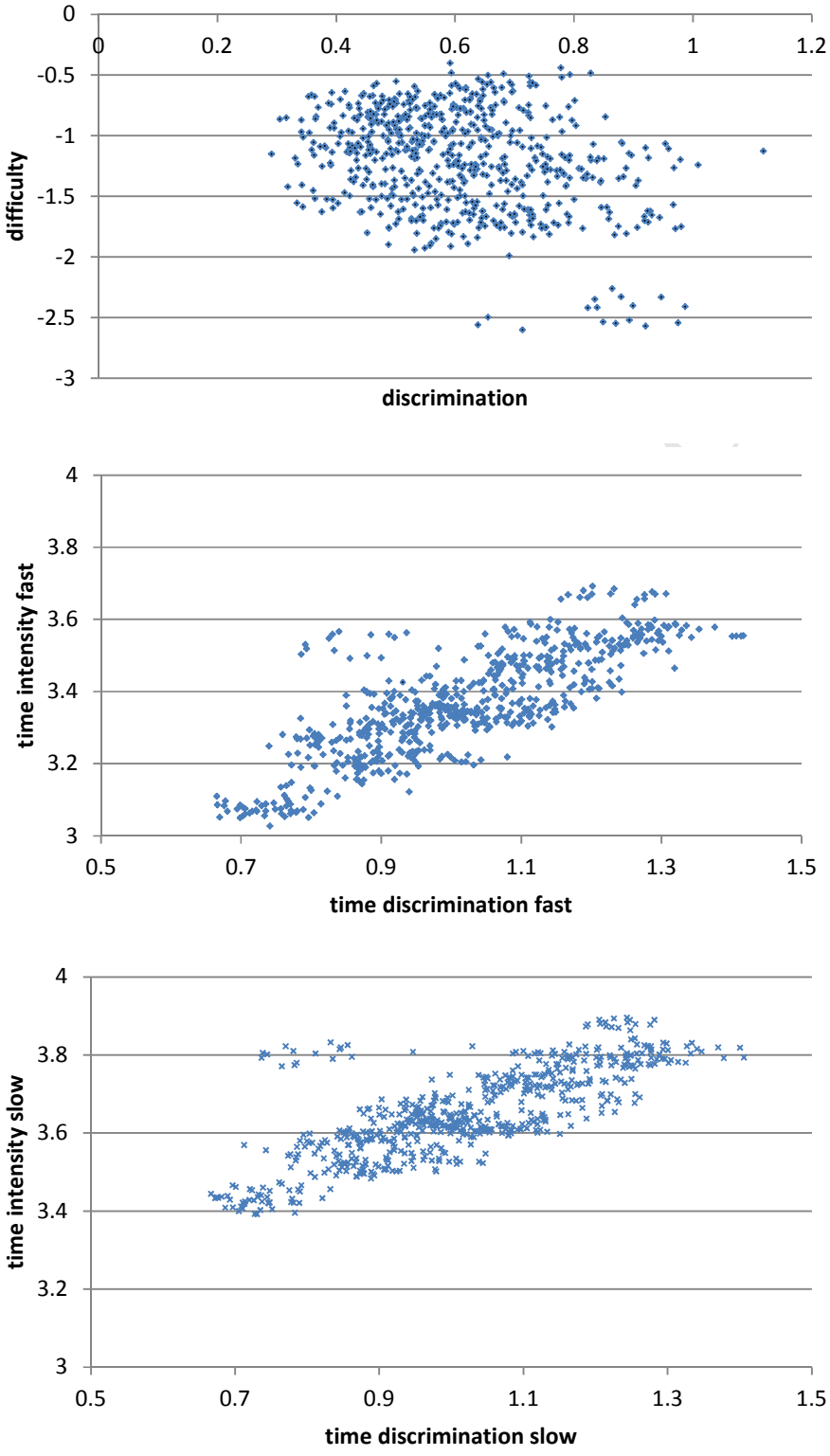


FIGURE 1. *Item and RT parameters of the simulated item bank*

Settings and Results of the Studies

The settings of the studies were based on those of the BSE. Test length was set equal to 40 items, and the maximum expected RT was set equal to 1,200 seconds. For an operational BSE consisting of 40 items, it turned out that slow test takers needed on average 400 seconds more than fast test takers (1,550 seconds compared to 1,150 seconds). As a consequence, most of the fast test takers and only few of the slow test takers were expected to meet this time limit. Test information was maximized for $\theta \in \{-2, -1.5, ..., 2\}$. A maximin approach (van der Linden & Boekkooi-Timminga, 1989) was applied to formulate the test assembly model. With this approach, the objective function of the test assembly problem can be formulated as:

$$\max \min_{\theta_p \in \{-2,-1.5,...,2\}} \sum_{i=1}^{I} I_i(\theta_p) x_i. \tag{24}$$

No weighting of various ability points was used. Software R (CRAN, 2014) with the lpsolve package (version 5.6.13) was used in this simulation study for all simulation conditions.

Study 1

In the first simulation study, the stochastic programming strategy was illustrated. In this method, the largest acceptable expected violation parameter β had to be defined. Unfortunately, there is no analytical method available for selecting this parameter. In this simulation study, several values of β were implemented.

First of all, a lower bound for the β parameter can be found by noticing that $\beta = 0$ implies that no violation is allowed. An upper bound for β can be identified as well. As was already mentioned in the settings of the study, slow test takers on average needed 400 seconds more than fast test takers, therefore $\beta = 400$ implies that even most of the slow test takers would be expected to meet the time limit constraints. As a result, reasonable values for the β parameter were expected to be in the interval $\beta \in [0,400]$, in this example. Since the 15% of the test takers were in the slow category, $\beta = 60$ (15% of 400) seemed to be a reasonable value for β. For illustrative purposes, a slightly stricter $\beta = 30$ and a less strict value of $\beta = 200$ were implemented as well. Finally, for larger β values, for example $\beta = 1000$, the time limit constraint does not affect item selection anymore. This value was added as a yardstick.

Results from Study 1

The resulting test information functions are shown in Figure 2. The maximin objective function in (24) maximized the minimum value of the test information function over all $\theta \in \{-2, -1.5, ..., 2\}$. For this data set, the minimum values of the information function over this interval were obtained for $\theta = 2$. At this ability level, for the condition with $\beta = 0$, where no violation of the time limit was allowed, test information at $\theta = 2$ was at its minimum compared to other settings for β (see Figure 2). By increasing the largest acceptable expected violation parameter β, the values of the resulting test information function at $\theta = 2$ also increased. Finally, for ($\beta = 200$), ($\beta = 400$), and ($\beta = 1000$), the test assembly process resulted in identical test information functions. Closer examination of the results revealed that the resulting tests for the conditions where $\beta \geq 200$ consisted of the same items. Item overlap between test forms resulting from various settings for the β parameter are shown in Table 1.

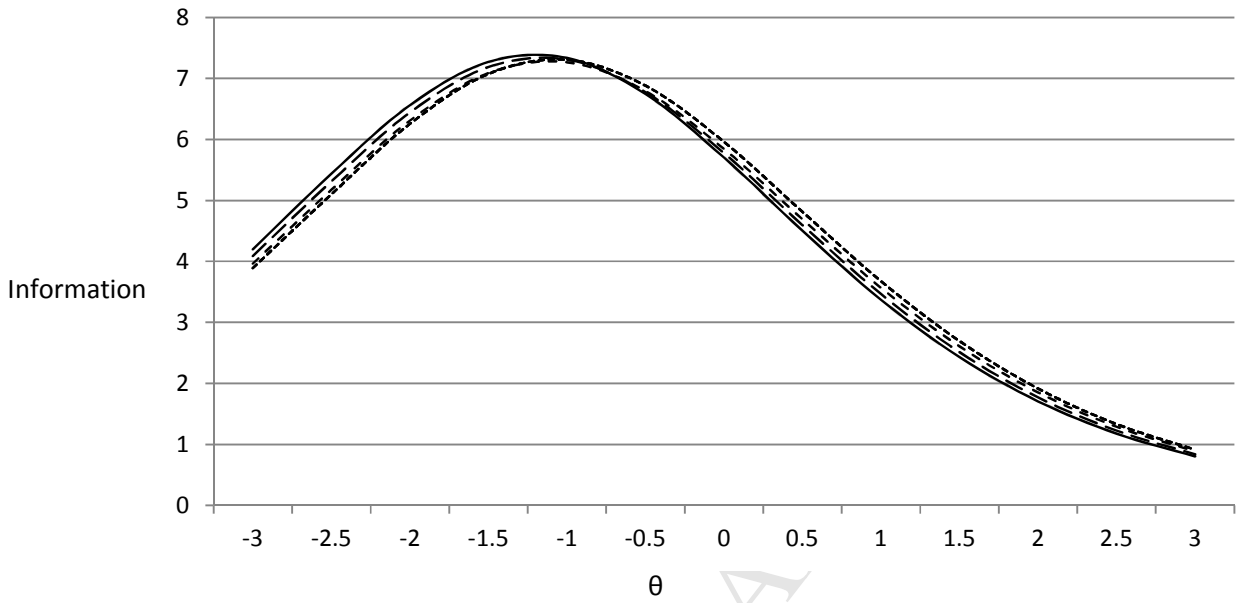


Figure 2. Test information function for various implementations of stochastic programming (β=0) solid, (β=30) large dashes, (β=60) medium dashes, (β=200), (β=400), and (β=1000) small dashes.

TABLE 1

Overlapping number of items for various values of the β parameter.

β parameter	0	30	60	200	400	1000
0		35	32	32	32	32
30			37	35	35	35
60				37	37	37
200					40	40
400						40
1000						

Study 2

In order to judge the attributed value of stochastic programming, a test was assembled using various strategies. The RT constraint for a mixed RT model distinguishing two classes of responses was modeled either based on average expected RTs (see Equation (15)), via a series of deterministic constraints, see Equation (16), as a robust optimization problem, see Equation (17), or by using stochastic programming. Given the settings of this study and the difference in time intensity between both classes of test takers, parameter β in Equation (23) was set equal to 60.

Results from Study 2

The resulting test information functions for Study 2 are shown in Figure 3. For $\theta \in \{-2, -1.5, ..., 2\}$, the minimum test information was maximized. With this implementation of the maximin method, the ability value $\theta = 2$ turns out to be most critical with respect to the maximin objective function. This is in line with our expectations, because the item bank was designed to provide a lot of information at the lower ability values. As a consequence, the items are not very informative at the higher ability values. Maximizing the minimum amount of information over $\theta \in \{-2, -1.5, ..., 2\}$, therefore tends to focus at the high ability levels. With respect to the various strategies, it can be seen that the second strategy, where the probabilistic constraint is replaced by a series of deterministic constraints, is much more conservative than the others. The robust ATA strategy is slightly more conservative than the strategy based on expected RTs. The stochastic programming strategy provides the most informative test with respect to the maximin objective function, in this example.

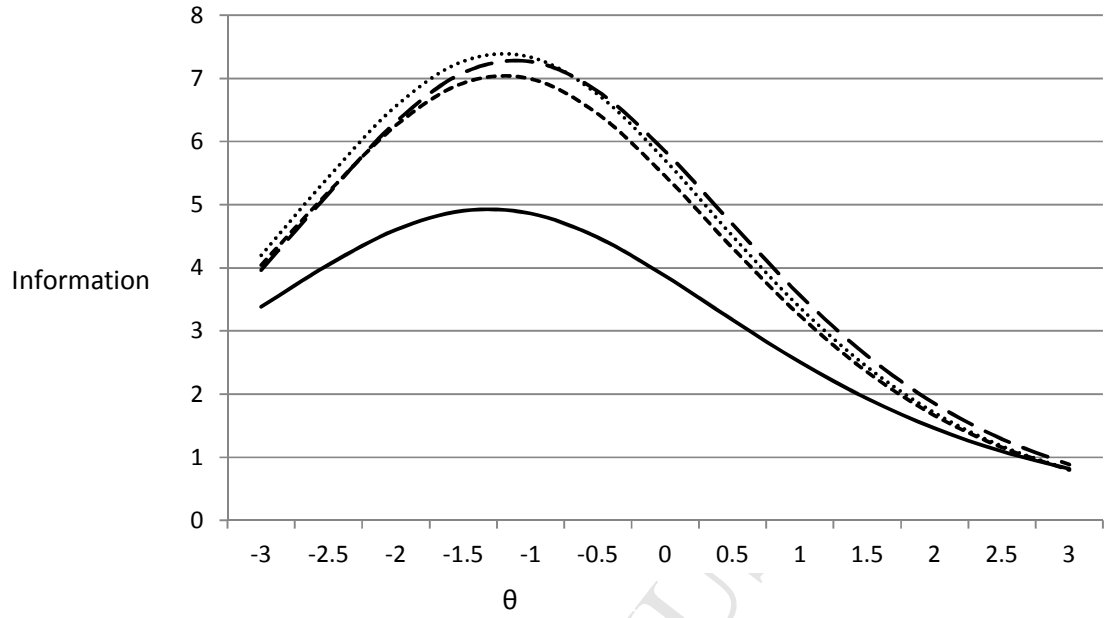


FIGURE 3. *Test information function for the various strategies: expected RTs (dotted), series of constraints (line), robust ATA (short dashes), stochastic programming (long dashes)*

Besides comparing the strategies with respect to the test information curves, the item overlap between various test assembly strategies was also calculated. The resulting overlap can be found in Table 2. As can be seen, the overlap in items between the second strategy (where the probabilistic constraint is replaced by two deterministic constraints) and the other strategies is very small. The test resulting from this strategy differs from all the other tests by more than 75% of the items. The difference between the other strategies is smaller. For example, tests assembled with the stochastic programming strategy and the expected RT strategy have 34 of 40 items in common. Differences between the robust ATA strategy and the stochastic programming strategy are larger: they only have 27 of 40 items in common.

TABLE 2

Overlapping number of items for various test assembly strategies

Strategy	Series of Constraints	Robust ATA	Stochastic
Expected RT	8	32	34
Series of Constraints		10	5
Robust ATA			27

Discussion

Computer-based assessments provide log files that may reveal more detailed information about the behavior of the test taker during test administration. In this paper, we focus on response time data. This kind of data could provide information about the average speed of working, about whether the test taker speeded up towards the end of the test, about warm-up effects, or about fatigue. Moreover, it could reveal information about whether the candidate was working at a stable speed or when different groups of test takers show different types of response behavior. The purpose of this research was to take variations in working speed into account and to introduce a new method for dealing with mixture RT constraints in automated test assembly.

Mixture RT constraints fall under the category of probabilistic constraints. During test assembly, the distributions of the item parameters are known, but, at the individual level, the RT parameters depend on the class membership of the test taker, which is a priori unknown. Nowadays, 0-1 LP methods are generally applied for solving test assembly problems. But these methods can only handle deterministic constraints, where the contribution of an item to a

constraint is fixed. Four strategies for handling probabilistic constraints in test assembly were introduced. Three of them reformulated deterministic alternatives to probabilistic optimization problems, while the fourth utilized stochastic programming.

Two simulation studies were conducted. The first simulation study illustrated the effects of various settings of the stochastic programming strategy. Implementing the stochastic programming method turned out to be rather straightforward when integrated chance constraints were applied. The only complicated part was that upper bounds β had to be derived for the chance constraints in (23). These upper bounds β indicate the largest acceptable expected violation. In other words, the probability of constraints being violated is limited by the upper bound β. The impact of the β parameter on the resulting test information functions was rather small, even though different β values resulted in selecting different items.

The second simulation study, comparing these four strategies, revealed that (a) replacing a mixture RT constraint with a series of RT constraints was far too conservative; (b) the robust ATA method performed only slightly worse than the method where the expected value of the RTs over the various classes was restricted; and (c) stochastic programming performed best for the test assembly problem in the example.

One of the biggest advantages of stochastic programming is that the probabilistic nature of the constraints is taken into account during test assembly. A disadvantage is that the models do not have the convenient properties that 0-1 LP models have when it comes to convexity of the solution space. Fortunately, several approximations have been proposed in the literature, and the approximation by using integrated chance constraints (Klein Haneveld & van der Vlerk, 2006) turned out to work well.

It should be mentioned that setting the value of β in the stochastic programming approach is a rather subjective process. In our second study, both response classes differed only with respect to the time intensity of the items. To set a value for β, we calculated the difference in average total RTs of a test of 40 items assembled from the simulated item bank for each class of response behavior. Given the prevalence of both classes, the percentage of test takers permitted to violate the RT constraint, and the skewness of the RT distribution, a setting of $\beta = 60$ seemed reasonable. In the case of a larger number of classes, or a greater number of differences between the classes, a different approach for selecting β would have to be applied. For example, when the mixture model described by Marianti et al. (2014) is applied, the RT behavior of the second class is unspecified. For this class of test takers, the average observed total RTs can be used as a guide for selecting β. For the mixture RT model of Molenaar and De Boeck (2014), the prevalence of slow and quick response behavior within one test taker could be used to obtain information about how to weight both classes for the whole population. Finally, in the case of dynamic RT models (Fox, 2014), one must take the expectation over the distribution of response behaviors, rather than a weighted combination of classes.

Overall, stochastic programming proved to be a useful strategy for the assembly of CBAs in the case of heterogeneous RT behavior of test takers. Several studies already demonstrated how heterogeneous RT behavior can be modeled by a mixture of lognormal RT models. These models have many advantages. They model RT behavior more accurately and can be used to detect aberrant response behavior with greater precision. Besides, they can be applied when the response behavior of test takers can be classified into a number of latent classes. Until now, the assembly of CBAs did not profit from these developments in the area of RT modeling. This

study illustrates how individualized CBAs can be assembled to provide more information about the test takers, even when uncertainty about their RT behavior has to be accounted for.

One remark must be made, however, with respect to the way the RT constraints were formulated in this study. The four different strategies focused only on the uncertainty due to the mixture of classes of response behavior. Uncertainties in RTs within each class were not taken into account. All of the constraints were formulated for the time intensities λ_i of the items, rather than for the RTs of the test takers. In order to formulate the constraints with respect to actual RTs, a two-level structure would have to be imposed. But since the purpose of this research was to introduce stochastic programming for dealing with mixture constraints, this additional source of uncertainty was not taken into account.

The next step in our research would be to implement stochastic programming in CAT and in multi-stage testing. In these modes of testing, information about the response behavior of the test taker becomes available during test administration, and it can be taken into account when selecting the next item or module. The shadow test approach (van der Linden, 2005) is very suitable for dealing with all kinds of constraints, and when it is combined with a stochastic programming method for solving item selection problems, it will be able to deal with probabilistic constraints related to, for example, mixture RT models as well. Another extension of this approach would be to take correlation between speed and ability into account. It should be noted thought that when stochastic programming is applied in the process of assembling of high-stakes CATs, the requirement of standardization of testing needs attention. The probabilities of the constraints will change for different test takers, based on more accurate information about their response behaviors.

Finally, in this research we focused on mixture RT constraints, and stochastic programming turned out to be a method that is suitable for dealing with these constraints. However, application of stochastic programming can easily be generalized to test assembly problems with, for example, mixture IRT models. For these problems, different classes of item information functions must be taken into account, and a probabilistic formulation of the objective function might have to be dealt with. Moreover, the application of stochastic programming could be generalized to any model that distinguishes latent classes of test takers during test assembly. What all of these applications have in common is that groups of test takers behave differently, and that the group to which a test taker belongs is unknown in advance. So whenever a test must be assembled for a heterogeneous population, stochastic programming might be considered as an alternative to the more restrictive 0-1 LP methods that are currently applied.

References

Ackerman, P. L., & Kanfer, R. (2009). Test length and cognitive fatigue: An empirical examination of effects on performance and test-taker reactions. *Journal of Experimental Psychology, 15*(2), 163–181.

Bertsimas, D., & Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical programming, 98,* 49–71.

Birge, J. R., & Louveaux, F. (1997). *Introduction to stochastic programming.* New York: Springer Verlag.

Cho, S. J., & Cohen, A. S. (2010). A multilevel mixture IRT model with an application to DIF. *Journal of Educational and Behavioral Statistics, 35,* 336–370.

Cohen, A. S., & Bolt, D. M. (2005). A mixture model analysis of differential item functioning. *Journal of Educational Measurement, 42,* 133–148.

CRAN (2014). *Statistical software package R (version 3.3.0).* Retrieved from http://cran.r-project.org/.

Egberink, I. J., Meijer, R. R., & Veldkamp, B. P. (2010). Conscientiousness in the workplace: Applying mixture IRT to investigate scalability and predictive validity. *Journal of Research in Personality, 44,* 232–244.

Evans, F., & Reilly, R. (1973). A study of speededness as a source of test bias. *Journal of Educational Measurement, 9,* 123–131.

Fan, Z., Wang, C., Chang, H. H., & Douglas, J. (2012). Utilizing response time distributions for item selection in CAT. *Journal of Educational and Behavioral Statistics, 37,* 655–670.

Finkelman, M. D., Kim, W., Weissman, A., & Cook, R. J. (2014). Cognitive diagnostic models and computerized adaptive testing: Two new item-selection methods that incorporate response times. *Journal of Computerized Adaptive Testing, 2,* 59–76.

Fox, J.-P., Klein Entink, R., & van der Linden, W.J. (2007). Modeling of responses and response times with the package CIRT. *Journal of Statistical Software, 20,* 1–14.

Fox, J.-P. (2014). *Modeling differential working speed in assessment testing.* (LSAC Report Series, RR 14-05). Newtown, PA: Law School Admission Council.

Glas, C. A.W. (2010). *Preliminary manual of the software program Multidimensional Item Response Theory (MIRT).* University of Twente, Enschede, The Netherlands.

Greiff, S., Niepel, C., Scherer, R., & Martin, R. (2016). Understanding students' performance in a computer-based assessment of complex problem solving: an analysis of behavioral data from computer-generated log files. *Computers in Human Behavior*, *61*, 36-46.

Hancock, G. R., & Samuelsen, K. M. (Eds.). (2008). *Advances in latent variable mixture models.* Charlotte, NC: IAP.

He, Q, & von Davier, M. (2014). *Extracting sequence patterns from process data of problem solving items using n-grams* . Paper presented at the 30th Workshop on IRT and Educational Measurement, Enschede, The Netherlands, November 19–21, 2014.

Hornke, L. F. (1997). Untersuchung von Itembearbeitungszeiten beim computergestützten adaptiven Testen. *Diagnostica, 43,* 27–39.

Klein Entink, R. H., Fox, J.-P., & van der Linden, W. J. (2009). A multivariate multilevel approach to the modeling of accuracy and speed of test takers. *Psychometrika, 74,* 21–48.

Klein Haneveld, W. K., & van der Vlerk, M. H. (1999). Stochastic integer programming: General models and algorithms. *Annals of Operations Research, 85,* 39–57.

Klein Haneveld, W. K., & van der Vlerk, M. H. (2006). Integrated chance constraints: Reduced forms and an algorithm. *Computational Management Science, 3,* 245–269.

Lawrence, I. (1993). The effect of test speededness on subgroup performance. Research Report 93–49, Princeton, NJ: Educational Testing Service.

Lee, Y. H., & Jia, Y. (2014). Using response time to investigate students' test-taking behaviors in a NAEP computer-based study. *Large-Scale Assessments in Education, 2,* 1–24.

Levine, M. V., & Rubin, D. B. (1979). Measuring the appropriateness of multiple-choice test scores. *Journal of Educational and Behavioral Statistics, 4,* 269–290.

Marianti, S. (2015). *R-package LNIRT.* University of Twente.

Marianti, S., Avetysian, M., Fox, J.-P., & Veldkamp, B.P. (2014). *Testing for aberrant behavior in response time modeling.* (LSAC RR 14-02).

Maris, E. (1993). Adaptive and multiplicative models for gamma distributed variables, and their application as psychometric models for response times. *Psychometrika, 58,* 445–469.

Masters, G., & Keeves, J. (1999). *Advances in measurement in educational research and assessment.* Amsterdam: Elsevier Science.

Molenaar, D., & de Boeck, P. (2014). *Response mixture IRT modeling of the speed accuracy trade-off in psychometric tests.* Paper presented at the 30th Workshop on IRT and Educational Measurement, Enschede, The Netherlands, November 19–21, 2014.

Muthén, L. K., & Muthén, B. O. (2012). Mplus. *The comprehensive modelling program for applied researchers: User's guide, 5*.

OECD (2015). *Students, Computers and Learning: Making the Connection*. PISA: OECD Publishing.

Partchev, I., & De Boeck, P. (2012). Can fast and slow intelligence be differentiated? *Intelligence, 40,* 23–32.

Schnipke, D. L., & Scrams, D. J. (1997). Modeling item response times with a two-state mixture model: A new method of measuring speededness. *Journal of Educational Measurement, 34,* 213–232.

Timmers, C., Walraven, A., & Veldkamp B.P. (2015). The effect of regulation feedback in a computer-based formative assessment on information problem solving. *Computers and Education, 87,* 1-9.

van der Linden, W. J. (2005). *Linear models for optimal test design*. New York: Springer.

van der Linden, W. J. (2006). A lognormal model for response times on test items. *Journal of Educational and Behavioral Statistics, 31,* 181–204.

van der Linden, W. J. (2007). A hierarchical framework for modeling speed and accuracy on test items. *Psychometrika, 72,* 287–308.

van der Linden, W. J. (2009). Conceptual issues in response-time modeling. *Journal of Educational Measurement, 46*(3), 247–272.

van der Linden, W. J. (2011). Test design and speededness. *Journal of Educational Measurement, 48,* 44–60.

van der Linden, W. J., & Boekkooi-Timminga, E. (1989). A maximin model for IRT-based test design with practical constraints. *Psychometrika, 54,* 237–247.

van der Linden, W. J., & Guo, F. (2008). Bayesian procedures for identifying aberrant response-time patterns in adaptive testing. *Psychometrika, 73,* 365–384.

van der Linden, W. J., & Reese, L. M. (1998). A model for optimal constrained adaptive testing. *Applied Psychological Measurement, 22,* 259–270.

van der Linden, W. J., Scrams, D.J., & Schnipke, D.L. (1999). Using response-time constraints to control for differential speededness in computerized adaptive testing. *Applied Psychological Measurement, 23,* 195–210.

van der Linden, W. J., & van Krimpen-Stoop, E. M. (2003). Using response times to detect aberrant responses in computerized adaptive testing. *Psychometrika, 68,* 251–265.

Veldkamp, B. P. (2013). Application of robust optimization to automated test assembly. *Annals of Operations Research, 206,* 595–610.

Veldkamp, B.P. (2016). On the Issue of Item Selection in Computerized Adaptive Testing With Response Times. *Journal of Educational Measurement, 53,* 212-228. .

Vermunt, J. K., & Magidson, J. (2013). *Latent Gold 5.0 Upgrade Manual.*

Wechsler, D. (2003). *Wechsler intelligence scale for children–Fourth Edition (WISC-IV).* San Antonio, TX: The Psychological Corporation.

Highlights of the paper:

1. The use of response times in the assembly of computer-based assessments (CBAs) was discussed
2. Various strategies for dealing with probabilistic constraints in CBAs were studied
3. Stochastic programming was introduced as a new method for CBA assembly
4. A simulation study provided insights in the strengths and weaknesses.
5. Stochastic programming turned out to be a valuable new tool