



# Review of the shadow-test approach to adaptive testing

Wim J. van der Linden<sup>1</sup>

Received: 9 August 2021 / Accepted: 20 August 2021 / Published online: 12 September 2021  
© The Author(s) 2021

## Abstract

Constrained adaptive testing is reviewed as an instance of discrete maximization with the shadow-test approach delivering its solution. The approach may look counterintuitive in that it assumes sequential assembly of full test forms as its basic operation. But it always produces real-time solutions that are optimal and satisfy the set of specifications in effect for the test. Equally importantly, it can be used to run testing programs with different degrees of adaptation for the same set of specifications and/or as a tool to manage programs with simultaneous processes as adaptive item calibration, time management, and/or item-security monitoring.

**Keywords** Adaptive item calibration · Adaptive testing · Automated test assembly · Discrete optimization · Item response theory · Item and test security · Mixed integer programming

## 1 Introduction

The distinctive feature of adaptive testing is sequential optimization of item selection. After each new response, the test taker's ability estimate is updated and the next item is selected to be optimal at the new estimate. Unlike fixed-form testing with its traditional paper-and-pencil format, adaptive testing assumes electronic delivery of the items with access to enough computational power to update the estimate and select the items. Thanks to the arrival of the personal computer as well as the recent introduction of cloud-hosted services, adaptive testing is now widely practiced in programs of psychological, educational, certification, and admission testing but also in such areas as marketing research and patient-reported outcome measurement.

A necessary condition for successful adaptive testing is a psychometric model with separate parameters for the properties of the items and the abilities of the test

---

Communicated by Maomi Ueno.

---

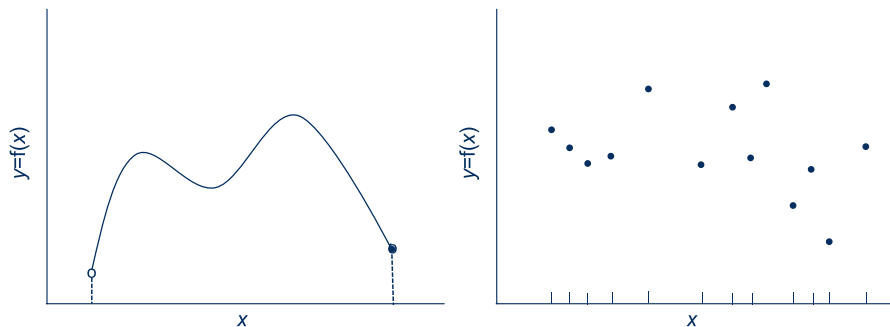
✉ Wim J. van der Linden  
wjvd Linden@outlook.com

<sup>1</sup> University of Twente, Enschede, The Netherlands

takers. Item response theory (IRT) offers a multitude of such models along with well-developed methods for their statistical treatment. Under each of these models, given a well-designed item pool, the process of adaptive testing reduces to the mathematical optimization problem of finding the sequence of items with combinations of parameter values optimally matching the sequence of the ability parameter updates. Criteria of optimization popular among the current adaptive testing programs are variations of Fisher's classical information measure. The measure, defined in (4) below, represents the statistical information about the ability parameter in the responses to the items as a mathematical function of the values of their parameters.

As the function is continuous in the item parameters, it may be tempting to think of the general problem of test assembly as an instance of the type of function optimization taught in the typical calculus class. This is not correct though. The optimization is not over the continuous space of the possible item parameter values but the discrete set of function values at the test taker's ability parameter across the items in the pool. More specifically, if the pool has  $I$  items, any test, no matter its format, needs to be assembled making acceptance–rejection decisions about the value of each of the items. In total,  $I$  binary variables are required to represent the decisions, which together imply optimization over a space of  $2^I$  possible outcomes.

The general distinction between function optimization in calculus and discrete optimization is illustrated using the two plots in Fig. 1. The method of finding the global maximum of an arbitrary real-valued function  $f(x)$  over the interval in the left-hand plot is to check all local maxima using first- and second-order derivatives and take the one with the greatest function value. If the interval is not open, the solution should be checked against the values of the function at its endpoints. Optimization over the discrete set of function values in the right-hand plot does not require any calculus but an enumerative algorithm that checks the function at each of the discrete points in its domain and retains the one with the largest value. For a function of a single variable, the time required for a direct check of all points may be manageable. But the computational complexity quickly increases with the number of variables defining the domain. In fact, the number of points that need to be checked grows exponentially with the number of variables. This is exactly the reason why test assembly with the Fisher information as objective function is an extremely



**Fig. 1** Example of real-valued function versus discrete maximization

complicated problem. As just noted, the maximization is then over a space of  $2^I$  points, a number that quickly increases with the number of items (for example, for an item pool with the modest size of 90 items, the number of points that needs to be checked is already larger than the size of the current world population).

For the problem of test assembly, the optimization problem is even more complicated than just the size of its solution space. The solution always has to satisfy a set of specifications in effect for the test. Examples of these specifications include the length of the test, tables of content specifications, desired distributions of item formats, numbers of items per set with the same stimulus, word counts, answer key distributions, etc. Each of these specifications needs to be imposed as a constraint on the selection of the items. The consequences for the optimization algorithm are enormous: rather than just enumerating all possible solutions until the one with the largest value for the information measure is found, it now also needs to check each of the possible solutions for its feasibility with respect to the constraint set. The checks need to be performed at the level of the complete test against the complete set of constraints. If an incomplete test would be checked, there is no guarantee whatsoever that a complete version of it would still satisfy the required set of constraints. Likewise, leaving out any constraints during earlier checks is bound to lead to failures at later checks.

It is exactly at this point that the switch to adaptive testing seems to run into unsurmountable complications. As each item already administered is automatically fixed, there is no guarantee whatsoever that the test could be continued without violation of some of the constraints and/or the selection of less favorable items. As an example, consider the case of a simple mathematics tests with algebra and geometry items and constraints in the form of lower and upper bounds on the numbers of items from each of the two content categories. Initially, items from both categories could be chosen. But if, for instance, the number of algebra items reaches its upper bound, the testing algorithm may be forced to pick items from the other category less than optimal at the current ability update. Now, imagine a more realistic mathematics test which requires control of its composition with respect to a more detailed set of content categories, different item formats, cognitive levels, word counts, answer key sequences, expected response times, and the presence of sets of items with a common stimulus or items that exclude each other. The combinatorial complexity of the problem quickly increases with each additional constraint necessary to deal with these specifications. All modern algorithms of constrained combinatorial optimization have a feature known as backtracking. If they get stuck, as in the examples of this fictitious mathematics test, they return to an earlier point to pursue an alternative route. But, unlike the assembly of fixed forms of a test prior to their administration, for adaptive testing this type backtracking is impossible. We just cannot apologize to a test taker and replace some of his/her earlier items when the algorithm gets stuck.

The mathematical optimization problem met in adaptive testing is, thus, an instance of constrained discrete optimization with a solution that has to be found sequentially without any backtracking to earlier items. On top of that, the algorithm needs to produce each next item without any latency noticeable to the test taker. In

spite of these challenges, the solution can be found by a simple twist of the problem of assembling fixed test forms, for which the methodology of mixed integer programming (MIP) already has proven to be a powerful resource. In the following, we first introduce the response model and optimization criterion used in our treatment of adaptive testing, briefly review the application of the MIP methodology to the problem of optimal fixed-form assembly, introduce the notion of a shadow test as the twist necessary to optimize constrained adaptive testing, and show how the result can be generalized to run testing programs with different degrees of adaptation for the same set of specifications and/or as a tool to manage programs with additional, simultaneously running processes such as adaptive item calibration, time management, and/or item-security monitoring.

## 2 Choice of model and item-selection criterion

The following notation is used. The items in the pool are denoted as  $i = 1, \dots, I$  while  $k = 1, \dots, n$  is the index for the positions for the items available in an adaptive test of length  $n$ . The event of item  $i$  assigned to position  $k$  is notated as  $i_k$ . Suppose  $k - 1$  items have already been administered to the examinee and  $S_{k-1}$  is the set of indices of these items. The next item in the test has to be selected from  $R_k = \{1, \dots, I\} \setminus S_{k-1}$ . As the case of arbitrary test takers is assumed, it is not necessary to introduce a separate index for them.

For dichotomous items, the distribution of response  $U_i$  for a test taker on item  $i$  is Bernoulli with probability mass function (pmf)

$$f(u_i; \pi_i) = \pi_i^{u_i} (1 - \pi_i)^{1-u_i}, \quad (1)$$

where  $\pi_i$  is the probability of a correct response for the test taker. The probabilities are assumed to follow the well-known three-parameter logistic (3PL) model

$$\pi_i \equiv c_i + (1 - c_i) [1 + \exp(-a_i(\theta - b_i))]^{-1}, \quad (2)$$

where  $\theta \in \mathbb{R}$  is the ability parameter of the test taker,  $b_i \in \mathbb{R}$  and  $a_i \in \mathbb{R}^+$  are parameters for the difficulty and discriminating power of the item, and  $c_i \in (0, 1)$  is the height of a lower asymptote to the response probability adopted to represent the effects of guessing. For notational convenience, we will use  $\xi_i \equiv (a_i, b_i, c_i)$  and represent  $\pi_i$  alternatively as  $\pi(\theta, \xi_i)$ . The adoption of the 3PL model is for presentation purposes only. As already noted, any model with separate item and test taker parameters explaining the response distributions on the items can be used to run an adaptive test. All items in the pool are assumed to be calibrated prior to operational testing. The only parameter with estimates that needs to be updated during testing is the test taker's ability parameter  $\theta$ .

The common item-selection criterion in adaptive testing is maximization of the information about  $\theta$  in the test taker's responses taken as a mathematical function of the item parameters. A measure known as the observed information is defined as

the negative of the curvature of the loglikelihood at  $\theta$  given response  $U_i = u_i$  on the item,

$$J_{u_i}(\theta; \xi_i) \equiv -\frac{\partial^2}{\partial \theta^2} \ln f(u_i; \theta, \xi_i). \tag{3}$$

Fisher’s information is the expected value of (3) across random responses

$$I(\theta; \xi_i) \equiv \mathcal{E}J_{U_i}(\theta; \xi_i). \tag{4}$$

For the 3PL model, the measure can be shown to have the closed-form expression

$$I(\theta; \xi_i) = a_i^2 \frac{1 - \pi(\theta; \xi_i)}{\pi(\theta; \xi_i)} \left( \frac{\pi(\theta; \xi_i) - c_i}{1 - c_i} \right)^2. \tag{5}$$

As both the ability and the item parameters are unknown, the typical frequentist approach is to evaluate (5) at point estimates substituted for each of the parameters, using

$$i_k \equiv \arg \max_{i \in R_k} \{I(\hat{\theta}_{k-1}; \hat{\xi}_i)\} \tag{6}$$

as criterion for the selection of the next item in the test.

A more honest approach is to select the items accounting for the remaining uncertainty about each of the parameters. A natural way of doing so is through sequential application of Bayes theorem. Let  $f(\theta | \mathbf{u}_{k-1})$  be the probability density function (pdf) for the posterior distribution of  $\theta$  after  $k - 1$  items have already been administered. As the items are assumed to have been calibrated before operational testing, the joint posterior pdf of each item is notated ignoring the calibration data, as  $f(\xi_i)$ . Since the two posterior distributions are independent, the update of the posterior pdf of  $\theta$  after administration of candidate item  $i \in R_k$  would be

$$f(\theta | \mathbf{u}_k) = \frac{\int f(u_i | \theta, \xi_i) f(\theta | \mathbf{u}_{k-1}) f(\xi_i) d\xi_i}{\int \int f(u_i | \theta, \xi_i) f(\theta | \mathbf{u}_{k-1}) f(\xi_i) d\theta d\xi_i}, \tag{7}$$

where  $f(u_i | \theta, \xi_i)$  is the model probability of observing  $U_i = u_i$  for the test taker and item. The posterior expected version of the criterion of maximum Fisher information in (5) now follows as

$$i_k = \arg \max_{i \in R_k} \left\{ \int \int I(\theta; \xi_i) f(\theta | \mathbf{u}_k) f(\xi_i) d\theta d\xi_i \right\}. \tag{8}$$

Calculation of (7) and (8) requires integration over the four unknown parameters driving the test taker’s response to the item. A convenient way of doing so is through Monte Carlo integration using a special implementation of the Gibbs sampler. The sampler cycles between (1) resampling of a sample from the posterior distributions of the item parameters permanently stored in the system and (2) a Metropolis–Hastings (MH) step for the ability parameter using prior and proposal distributions conveniently calculated from the previous posterior distribution. Let  $(\xi_i^{(1)}, \dots, \xi_i^{(S)})$  be a

vector of draws for the parameters of item  $i$  sampled from the vector stored in the system and  $(\theta^{(1)}, \dots, \theta^{(S)})$  a vector of draws of the test taker's ability parameter  $\theta$  produced by the MH step (the assumption of equal vector length is for notational convenience only). The next item is then selected calculating the criterion of the maximum posterior expected information in (8) as

$$i_k \equiv \arg \max_{i \in R_k} \left\{ S^{-1} \sum_{s=1}^S I(\theta^{(s)}; \xi_i^{(s)}) \right\}, \quad (9)$$

that is, simply as the average of the Fisher information in (5) across the last posterior draws of each of the parameters currently present in the system. For further details of this extremely fast sampler, the reader is referred to van der Linden and Ren (2020).

### 3 Fixed-form assembly

We discontinue our treatment of adaptive testing temporarily to focus on the simpler problem of assembling an optimal fixed test form prior to its administration.

It is tempting to think of test assembly as a unique problem that requires its own special algorithm to produce the solution, but this is incorrect. Since a seminal paper by Theunissen (1985), we are aware of the fact that it can be conducted with optimal results as an application of the methodology of mixed integer programming (MIP) (e.g., Chen et al. 2010; Williams 1999). The application involves four distinct steps: (1) choice of decision variables; (2) use of the variables to model both the objective function that needs to be optimized and the constraint set; (3) a call to a MIP solver; and (4) the evaluation of the solution returned by the solver. Use of the MIP methodology thus separates the formulation of the test specifications from the algorithm that searches for the best collection of items for the test. The basic solution produced by a solver is a string of values for the decision variables indicating which items should be selected and which should not. Powerful MIP solvers are readily available in the form of standard software packages. When, for some reason, inspection of the solution reveals that an extra constraint should be added to the model or some of the bounds should be changed, it is simple to edit the model and resubmit it to the solver.

The basic decision variables are binary variables representing the acceptance–rejection decision for each of the items in the pool:

$$x_i = \begin{cases} 1, & \text{item } i \text{ is accepted,} \\ 0, & \text{item } i \text{ is rejected,} \end{cases}$$

$i = 1, \dots, I$ . For technical reasons, an occasional real-valued variable may be needed, hence the name of mixed integer programming. Figure 2 shows the setup for an item pool of 100 items. Each of the rows identifies a different possible form in case no test specifications would have to be satisfied. The total number of forms for this modest size item pool is, thus, already  $2^{100}$ .

Item	1	2	...	$i$	...	99	100
Variable	$x_1$	$x_2$	...	$x_i$	...	$x_{99}$	$x_{100}$
Form 1	0	0	...	..	...	0	0
Form 2	0	0	...	...	...	0	1
Form 3	0	0	...	...	...	1	1
...	...	...	...	...	...	...	...
Form 2 <sup>100</sup>	1	1	..	...	...	1	1

Fig. 2 Decision variable for the selection of an unspecified test from a pool of 100 items

The variables are easily used to model the test specifications. Following is an example of a model for the assembly of a selection or admission test with a cutoff score of  $\theta_c$ . The test is required to satisfy the following specifications:

1. Maximum information at the cutoff score;
2. Test length equal to  $n$ ;
3. Upper and/or lower bounds  $n_c$  on the numbers of items in content categories with indices  $V_c$ ,  $c = 1, \dots, C$ ;
4. Upper and/or lower bounds  $b_q$  on the sum of quantitative attributes  $q_i$  of the items;
5. No items in the test that contain clues to each other (“enemy items”);
6. Items with indices in subset  $V_0$  in the pool excluded from the test;
7. Items with indices in subset  $V_1$  in the pool included in the test.

A model that delivers the desired form is

$$\text{maximize } \sum_{i=1}^I \left( S^{-1} \sum_{s=1}^S I(\theta_c; \xi_i^{(s)}) \right) x_i \tag{10}$$

subject to

$$\sum_{i=1}^I x_i = n; \tag{11}$$

$$\sum_{i \in V_c} x_i \leq n_c, \quad c = 1, \dots, C; \tag{12}$$

$$\sum_{i=1}^I q_i x_i \leq b_q, \quad q = 1, \dots, Q; \tag{13}$$

$$\sum_{i \in V_e} x_i \leq 1, \quad e=1, \dots, E; \quad (14)$$

$$\sum_{i \in V_0} x_i = 0; \quad (15)$$

$$\sum_{i \in V_1} x_i = n_1; \quad (16)$$

$$x_i \in \{0, 1\}, \quad i = 1, \dots, I. \quad (17)$$

The objective function guarantees maximum posterior expected Fisher information at  $\theta_c$  (now with averaging over the posterior distribution of the item parameters only). The first constraint fixes the length of the test. In (12) and (13),  $\leq$  should be taken to be the (strict) (in)equality necessary to represent the intended specification. The constraints in (12) can also be used to model specifications with other categorical item attributes as their format, presence of graphical material, answer key, etc. The constraints in (13) restrict the sum of quantitative item attributes across the test. Examples of such attributes include word counts, expected response times, and statistical item parameters. Constraints as in (14) prevent the selection of more than one item from enemy sets with their indices in subsets  $V_e$ ,  $e = 1, \dots, E$  in the pool. If certain items are rejected in advance, their indices are assumed to be collected in subset  $V_0$  in (15). On the other hand, when the choice of a subset of items (e.g., a set of anchor items) must be fixed in advance, (16) should be used. Finally, (17) constrains the decision variables to be binary.

The model in (10)–(17) represents just the core of a typical real-world test-assembly model. Additional constraints are likely to be necessary, for instance, to represent targets for test information functions stretching over multiple  $\theta$  values, logical structures such as items nested in sets with common stimuli, to match an existing form item by item, assemble a set of parallel forms, deal with different sections in the test, or format the form to prepare for administration. Also, the objective function and one of the constraints can be reformulated to exchange roles. For a comprehensive introduction to modeling test assembly, refer to van der Linden (2005, 2018a).

Observe that each of the mathematical expressions in (10)–(17) is linear in the decision variables. In fact, they take only two different forms, as a simple sum of the decision variables over a set of items in the pool or a weighted sum with quantitative attributes as coefficients. As other types of expressions have never appeared necessary to represent any test specification, it seems safe to conclude that linear MIP is enough to deal with all of real-world test assembly. Solvers for this linear type of optimization are available in commercial packages or as open source. Examples of the former are *CPLEX Optimizer* (IBM 2021), *FICO Xpress Solver* (FICO 2021), and *Gurobi* (Gurobi Optimization LLC 2021), whereas the latter include *GLPK* (GNU Linear Programming Kit 2021) and *lp\_Solve* (Berkeelaar et al. 2021). Commercial solvers are generally expensive but usually offer



academic licenses at nominal costs while, of course, open source solvers are for free. The R package *TestDesign* offers a convenient interface to several of these solvers, supports model formulation in the form of simple menu choices, and contains tools to visualize results (for an introduction, see Choi et al. (2021)).

In some circles, there still exists a preference for self-programmed heuristic algorithms because of an assumed advantage of their speed (van der Linden and Li 2016). But though MIP problems still should be considered as NP-complete, as a result of smart preprocessing of the problem, capitalization on special problem structures, efficient implementation of branch-and-cut methods, and options to stop with negligible loss of optimality, the development in the performances of their solvers over the last three decades or so have been more than dramatic. For example, Bixby (2012) demonstrated a machine-independent increase in performance on a set of 1,982 benchmark problems between the first release of *CPLEX* (1991) and *Gurobi* (2009) to be greater than a factor of 5,000,000. Similar conclusions have been reached by Koch (2011). Note that the time of these studies is more than a decade behind us while improvements have continued unabatedly.

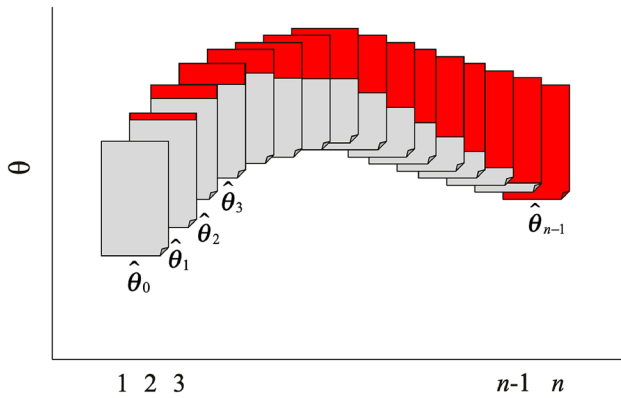
It is natural to assume that, though now indispensable for fixed-form test assembly, use of these solvers for the more difficult problem of adaptive assembly still is beyond reach. However, the contrary is true. As shown in the next section, the sequential nature of the application actually allows for much faster running times for a well-tuned solver resulting in item selection without any noticeable latency to the test taker.

## 4 Adaptive test assembly

The main obstacle that seems to block the application of MIP to adaptive testing is its inherent dilemma between sequential and simultaneous test assembly: In adaptive testing, the items are necessarily selected *sequentially* to allow for the ability updates whereas they must be selected *simultaneously* to check on their feasibility with respect to the constraint set. As backtracking to correct for earlier errors is impossible, the only solution left is to look forward, treating adaptive testing as selection from a sequence of fixed forms with an update of the ability estimate after each form. The test taker will never see any of these forms, hence their name of *shadow test* (van der Linden and Reese 1998; van der Linden 2005).

Figure 3 illustrates the setup of the shadow-test approach (STA), which cycles through the following steps:

1. The first shadow test is assembled to have maximum information at the initial ability estimate  $\hat{\theta}_0$  while satisfying the complete set of constraints.
2. The item from the shadow test with maximum information at  $\hat{\theta}_0$  is administered to the test taker; all items not administered are returned to the pool.



**Fig. 3** Diagram of the shadow-test approach to adaptive testing. Vertical axis represents the ability scale, horizontal axis the order of the items selected for administration. Red parts of the shadow tests represent the items already administered; gray parts remain hidden from the test taker

3. The two previous steps are repeated for subsequent items  $k = 2, \dots, n$  each time using the previous responses  $\mathbf{u}_{k-1} = (u_1, \dots, u_{k-1})$  to update  $\hat{\theta}_{k-1}$  and, as the objective function has changed, with the constraint

$$\sum_{i \in S_{k-1}} x_i = k - 1 \tag{18}$$

added to the model to include all items in the set  $S_{k-1}$  already administered in the next shadow test.

The final shadow test is the actual adaptive test. As each of the earlier shadow tests meets each of the constraints, the adaptive test automatically meets each of them. Likewise, as each shadow test is optimal at the ability update and only its optimal free item is administered, the adaptive test is always optimal given the set of constraints. (Of course, the best possible test is the one with all items maximally informative at the *true* value of the test taker’s ability parameter. It may still be possible to get closer to it, for instance, by using collateral information about the parameter to improve both on the initial estimate and its subsequent updates.)

The STA thus requires the assembly of as many fixed forms as items administered. It is possible to realize this without noticeable latency capitalizing on the specific sequential nature of adaptive testing. First of all, assuming the problem has been modeled efficiently and appropriate solver settings are used, the option exists to run the model prior to the start of the test and use the solution as the first shadow test. Secondly, once the test has started, subsequent shadow tests can be found using hot starts of the solver, that is, with the previous shadow test as initial solution for the next. The start forces the solver to begin the search for the optimum directly in the subspace of feasible solutions, generally an enormous time-saving step. [The extra constraint in (18) does not hurt in any way as a hot start implies decision variables of *all* items in the pool fixed at their value in the previous solution.] Thirdly, due to convergence of the ability estimates, the change in the value of the objective function quickly becomes minor. As a result, the shadow test typically does not

change at all for extended periods during the later part of the test; for options to capitalize on this observation, see Choi et al. (2016). Finally, unlike adaptive testing with heuristic algorithms, the STA selects the next item for administration directly from the free part of the shadow test rather than the entire pool, without any need to bother about constraint satisfaction. An empirical example of the STA with running times for the solver in milliseconds is presented below.

Because of its setup, the STA enables us to deal with any type of constraint met in fixed-form assembly. It can be used for much more though. Examples include item-exposure control during testing using random item-ineligibility constraints (van der Linden and Choi 2020), control of the differential speededness inherent in any type of adaptive testing (van der Linden and Xiong 2013), equating number-correct scores on adaptive tests to a fixed form for the same test (van der Linden, 2001), efficient assembly of item pools for adaptive testing (van der Linden et al. 2006), implementation of  $\alpha$ -stratification in adaptive testing (Chang and van der Linden 2003; van der Linden and Chang 2003), improved sequencing of an adaptive test battery (van der Linden 2010), running adaptive tests with items generated on the fly rather than selected from a fixed pool (Geerlings et al. 2013). In addition, it can be generalized to run tests with different adaptive formats for the same set of test specifications or manage testing programs with multiple adaptive processes. The last two applications are the subject of the next two sections.

## 5 Generalizing the STA

Once we have a shadow-test assembler in place, its functionality is easily extended to serve as a general tool for optimal test assembly. The options that exist are alternative objectives for the assembly of the shadow tests, alternative objectives for the selection of items from the shadow tests, as well as the options of varying the number of shadow tests per test taker or the number of test takers per shadow test. Choosing the right combination, tests can be assembled with nearly every format or degree of adaptation. The following review of options is derived from van der Linden and Diao (2014).

Alternative objectives for the shadow tests are:

1. maximization of Fisher's information at predetermined  $\theta$  values rather than ability estimates;
2. approaching a target for the test-information function at multiple  $\theta$  values, as in fixed-form assembly;
3. objective functions based on other item or test attributes than Fisher's information;
4. combinations of multiple objectives in a single function;
5. different choices of objectives at different moments during the test.

As demonstrated below, the combination of the first and last option allows us to implement multistage testing (MST) without the necessity to assemble different

subtests for the different stages in advance. A similar combination of objectives can also be used for adaptive testing for selection or admission decisions with a different cutoff score for the different treatments or receiving institutions. The adaptive test could then begin using maximum Fisher information in (9) at the ability updates as objective function but move to the function in (10) with the cutoff score for the treatment or institution closest to the current estimate later in the test. Combinations of different objectives in a single function belong to the domain of multi-objective test assembly for which several standard options are available (Veldkamp 1999). An example of an alternative attribute as objective is a test that minimizes the difference between the time limit and the predicted actual time. The objective requires simultaneous updating of the test takers' speed and ability parameters and could be introduced replacing the objective of maximum information toward the end of the test. Choi et al. (2021) show an example of adaptive testing with the same fixed target for the test-information function over a range of  $\theta$ s for each test taker.

It seems natural to assume the same objective of maximum information for the shadow tests as for the item selected from them, but this is not necessary either. Alternative item-selection objectives include:

1. objectives with respect to another item or test attribute;
2. objectives based on combinations of attributes;
3. different choices of attributes at different moments during the test.

The choice of a different attribute can be motivated by the fact that the shadow test already contains the currently best local selection of free items in the pool. If the pool is well designed, the possibility to optimize the test with respect to a second attribute may then be worth the loss involved in selecting a somewhat less informative item, especially toward the end of the test. The option could be used, for instance, to choose some of the relatively underexposed items for security reasons. In fact, alternating between different objectives for shadow-test assembly and item selection opens up an entirely new possibility of multi-objective test assembly not available for fixed-form assembly.

The standard approach has been to assemble a new shadow test prior to each item. However, the following alternatives exist:

1. assembling multiple shadow tests prior to each item;
2. keeping the same shadow test for sequences of items;
3. combinations of the two options or alternation between them at different moments during the test.

Assembling a set of test forms simultaneously requires only minor changes in the test-assembly model (van der Linden 2005, sect. 6.2). The option was used by Veldkamp and van der Linden (2008) to assemble sets of parallel shadow tests prior to the selection of each item with the goal to create larger sets of free items to pick from for the next item to administer, a condition known to improve the earlier Symson and Hetter (1985) method of random item-exposure control for adaptive testing.

The second option allows us to “freeze and thaw” shadow tests at different moments during the test. Different versions of the option have been used to study their effect on item-selection latency in adaptive testing (Choi et al. 2016). The option will also be used in our empirical example below to evaluate testing formats with different degrees of adaptation.

The last options vary the number of shadow test relative to the number of test takers. They include:

1. different shadow tests for each test taker;
2. the same shadow tests for multiple test takers;
3. combinations of both options.

The first option is used in the standard STA with updates of the ability estimates after each item but can also be used to adopt different strategies for different groups of test takers. The second option can be used to assemble a first shadow tests matching the information on the test taker’s ability available prior to the test which is then administered in its entirety (linear-on-the-fly testing).

Observe that each previous option is available no matter the constraint set for the test. van der Linden and Diao (2016) used the feature to determine the relative efficiencies of testing formats with different degrees of adaptation while maintaining exactly the same content for the test. To the best of the author’s knowledge, this type of comparison has never been conducted elsewhere, which is not surprising given the heuristic nature of several of the alternative algorithms still in use for adaptive testing.

The four formats with a decreasing degrees of adaptation were:

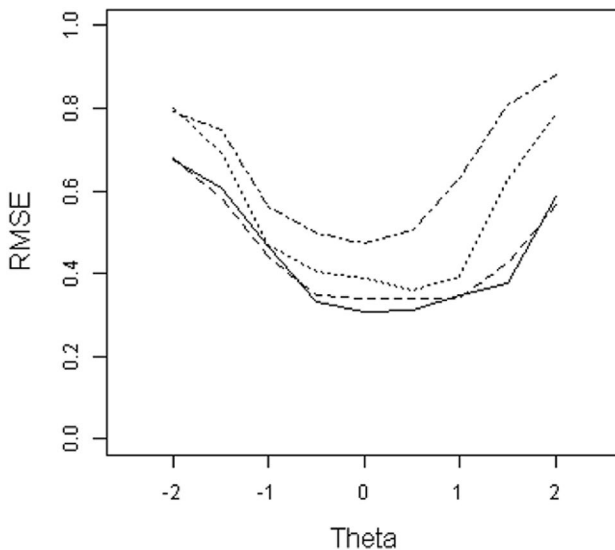
1. adaptive test;
2. on-the-fly MST with an adaptive routing test;
3. standard MST with fixed subtests per stage;
4. linear test.

The study was conducted simulating 30-item tests from a pool of 300 items randomly taken from an inventory for a real-world testing program. All items had been calibrated under the 3PL model. A total of 53 constraints was necessary to model the content specifications in use for the test. The objective both for the shadow tests and the selection of items from them was maximum Fisher information. The simulations were replicated for 250 test takers at  $\theta = -2.0, -1.5, \dots, 2.0$  each. The initial ability estimate for each testing format and simulated test taker was  $\hat{\theta}_0 = 0$ .

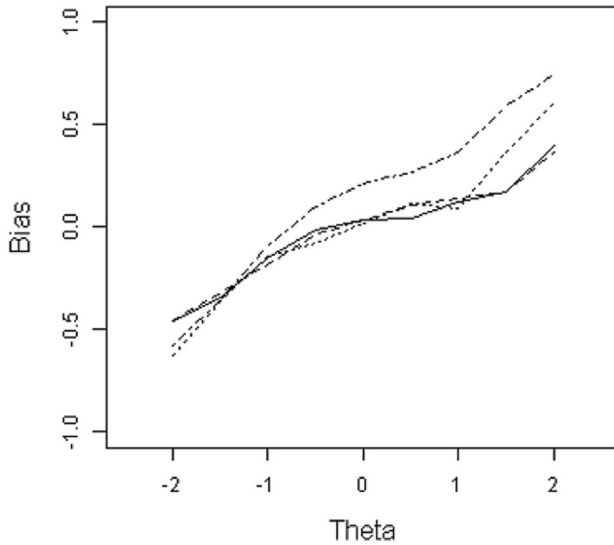
The fully adaptive version of the test was implemented using the standard STA with re-assembly of the shadow test after each next item. The linear test was pre-assembled using one run of the shadow-test assembler with a uniform target at maximum height for the test information function across  $\theta = -1.5, 0$ , and  $1.5$ . The target was realized applying the maximin criterion across the three ability levels (van der Linden 2005, p. 69–71). Both MSTs had a 1-3-3 format created using the earlier “freeze and thaw” mechanism with the shadow test re-assembled

at different choices of objective function. The subtests for the fixed MST were pre-assembled using seven runs of the shadow-test assembler, one for each of its possible routes. Each of the runs began with the assembly of the initial shadow test using the same maximin criterion across the same three ability levels as for the linear test. The shadow test was then re-assembled after 10 and 20 items at each of the combinations of the three ability levels for the seven routes. During the simulated test administrations the test takers were assigned to the subtests comparing their ability estimates with the midpoints between  $\theta = -1.5, 0,$  and  $1.5$  as cutoffs. The on-the-fly version of the MST was simulated freezing the shadow test for 10 items and then re-assembling it at the test taker's ability update.

In spite of their identical set of content constraints, the testing formats resulted in quite different root-mean-square-error (RMSE) and bias functions for the final ability estimates. As Figs. 4 and 5 show, not unexpectedly of course, the functions for the linear format were worst, while those for adaptive testing and MST on the fly were best. The minor differences between the latter confirm our earlier observation that, due to quick convergence of the ability estimates, the shadow test may not change much for extended periods, especially toward the end of the test. In spite of its broad-range routing test, the 1-3-3 MST format with fixed subtests with maximum information at three predetermined design points clearly led to considerable loss of quality of the final ability estimates. The RMSEs for the ability estimates at the high end of the scale were even closer to the linear version than the two adaptive versions of the test.



**Fig. 4** RMSE function for the final ability estimates after 30 items (solid curve: adaptive test; dashed curve: MST on the fly; dotted curve: standard MST; dotted-dashed curve: linear test)



**Fig. 5** Bias function for the final ability estimates after 30 items (solid curve: adaptive test; dashed curve: MST on the fly; dotted curve: standard MST; dotted-dashed curve: linear test)

## 6 STA as a management tool

Group-based, paper-and-pencil testing has its origin in the time about World War I. Its success has led to a testing industry with a well-established technology that has been dominant until today. One of its key characteristics is a cycle of test development consisting of the typical stages of item writing, field testing of the items, test-form assembly, test administration, data cleaning, item calibration, model fit analyses, linking of the items to a standard scale, forensic analyses, and test scoring. Each of these stages relies on output from the preceding stages and serves as input to the next. The process is known to create time pressure for testing programs with annual administration plans, tends to result in long waiting times for the scores to be released, and has high levels of vulnerability. A defective item not detected right away, a test form compromised during pretesting, or a statistical error made during parameter linking is bound to lead to patch work later on in the process or even to stages that must be redone completely. Needless to say that managing such processes can be a challenge, especially if the stakes for the participants are high.

The first testing programs that became adaptive were basically developed along the same lines. Though the items were written to fill a complete pool, they were field tested using paper forms, the item parameters were estimated and linked through complicated setups, and analyses to check on possible security breaches were performed post hoc. The only direct change was replacement of group-based sessions by individually scheduled tests with immediately available scores, for several testing program one of the main drivers to go adaptive.

Now that we have more experience with adaptive testing, the computational power of its infrastructure has increased, and the arrival of cloud-hosted services has

made test delivery possible across geographical borders, it has become clear that the traditional work flow might have become obsolete. In fact, we are already witnessing an emerging transition from the cyclical process of test development with its sequence of stages, analyses of massive data files, and use of stand-alone software programs to real-time operations that run each of the old stages simultaneously processing the test takers' responses one at a time.

The first sign of the transition was the addition of item-exposure control procedures to the process of adaptive testing, the reason being that, though no longer vulnerable to the threat of stolen test forms that had plagued paper-and-pencil testing, an extended use of the same item pool, in combination with its inherent tendency to capitalize on a small subset of highly informative items, had created the opportunity for test takers to memorize items and share them with future colleagues. The process of randomized item-exposure control discussed below runs simultaneously with the one of selecting the items for administration and has separate item parameters that require real-time updating. Another example is online item calibration with the embedding of a few newly written items in the adaptive test for each test taker and a real-time update of their parameters. Examples of other candidate processes include continuous item-fit monitoring, cheating detection, checks on item security, item-pool maintenance, time-on-test management and, ultimately, on-the-fly item generation.

Fortunately, four different tool kits are available to support the transitions:

1. Psychometric modeling of the interaction between the test takers and the items. An obvious example is the choice of the response model with its parameters for the test taker and the items. A model with a similar type of parameterization for the response times (RTs) on the items is necessary to monitor and control all time aspects of the test (for a candidate not reviewed in this article, see van der Linden 2016). Other probabilistic models may be needed to address specific processes, for example, the model that performs item-exposure control with random item-ineligibility constraints reviewed below
2. Statistical methods for sequential parameter updating. An important distinction is the one between intentional and nuisance parameters in each of the models. Both types of parameters are necessary to make the models realistic. In plain adaptive testing, the ability parameters are intentional and consequently require an update after each new item. The item parameters, on the other hand, are nuisance parameters; they are required to select the items and score the test takers but, other than that, are not of interest. For the process of adaptive calibration of new items, their roles are reversed. The focus then is entirely on the item parameters. The natural update of the intentional parameters is through sequential application of Bayes theorem with the type of Gibbs sampler already used to calculate the item-selection criterion in (9). The basic trick underlying its implementation is to resample the posterior samples for the nuisance parameters currently present in the system while performing an MH step for the intentional parameters.
3. Sequential updating of the parameter estimates is necessary but not sufficient to make each of the processes adaptive. To achieve adaptation, each update should be followed by an improved choice of the next datum to be collected. The field



of statistics that helps us to optimize the data collection is optimal design theory. One of its best criteria of optimality is the criterion of  $D$ -optimality, of which the Fisher information in (4) is an example for the case of a model with a single intentional but multiple nuisance parameters.

4. The shadow-test approach is available to manage all processes simultaneously. It was mainly introduced to update the ability parameters sequentially while keeping all non-statistical parameters of the test between their bounds, but its foundation in MIP as well as the generalizations discussed in the previous section make it perfectly suitable to manage multiple adaptive processes simultaneously. The next two examples are to illustrate the claim; a more in-depth introduction to this extended role of the STA, is available in van der Linden (2018b).

The basic idea underlying the item-exposure control method is to exclude items with a tendency of overexposure randomly from administration with probabilities that adapt themselves to the desired level of exposure. The random nature of the control is to prevent predictability of the items being seen by the test takers while the adaptation allows us to run the method without any need to supervise its operation. Though it is common to control the exposure rates conditional on the ability parameter, for notational simplicity, the version with the marginal rates introduced van der Linden and Veldkamp (2014) is reviewed (for an application of the conditional method, see Choi and Lim 2021).

Prior to the administration of the test to a new test taker, Bernoulli experiments are conducted for each item in the pool to determine its eligibility. Items that are ineligible are constrained out of the shadow tests for the test taker. When the test is completed, the probabilities controlling the experiments are updated to prepare for the next test taker. We use  $E_i$  to denote the event of item  $i$  being eligible for the test taker and  $A_i$  for its administration.  $Pr\{E_i\}$  is the probability of eligibility used in the Bernoulli experiments for the items, while  $Pr\{A_i\}$  is their exposure rate. The exposure rate is required to be below an upper bound  $r^{\max}$ .

The administration of an item implies its eligibility. It thus holds that

$$A_i \subset E_i, \quad \text{for each } i.$$

Consequently,  $Pr\{A_i\} = Pr\{A_i | E_i\}Pr\{E_i\}$  and the maximum exposure rate, thus, requires

$$Pr\{A_i | E_i\}Pr\{E_i\} \leq r^{\max}. \tag{19}$$

Alternatively, as  $Pr\{A_i | E_i\} = Pr\{A_i\}/Pr\{E_i\}$ , (19) can be rewritten as

$$Pr\{E_i\} \leq \frac{r^{\max}Pr\{E_i\}}{Pr\{A_i\}}, \quad Pr\{A_i\} > 0. \tag{20}$$

Let  $l = 1, 2, \dots$  denote the test takers in the order in which they arrive. The inequality in (20) can be conceived of as a recurrence relation giving us the necessary update of the eligibility probabilities from test taker  $l - 1$  to  $l$  as

$$\Pr^{(l)}\{E_i\} \leq \min \left\{ \frac{r^{\max} \Pr^{(l-1)}\{E_i\}}{\Pr^{(l-1)}\{A_i\}}, 1 \right\} \quad (21)$$

The probabilities in the right-hand side of (21) are updated using the counts of item eligibility and administration adjusted after each test taker. Observe that the update is adaptive. As soon as  $\Pr^{(l-1)}\{A_i\}$  happens to be greater than  $r^{\max}$ , the eligibility probability is adjusted downwardly whereas the upper bound on it is relaxed when it is smaller than  $r^{\max}$ .

An obvious way to constrain ineligible items out of the test is by collecting their indices in a set  $V_0$  and adding the constraint in (15) to the shadow-test model. However, in principle, if the set of ineligible items is too large for a test taker, the shadow-test model may become infeasible. Though this happens rarely for the case of a well-designed item pool with realistic target values for the exposure rates, the possibility does require attention. An effective solution is to extend the objective function in (10) with a penalty term:

$$\text{maximize } \sum_{i=1}^I \left( S^{-1} \sum_{s=1}^S I(\theta^{(s)}; \xi_i^{(s)}) \right) x_i - M \sum_{i \in V_0} x_i, \quad (22)$$

with  $M$  an arbitrary value chosen to be greater than the maximum Fisher information for every item in the pool. The penalty term replaces the ineligibility constraint in (15). Because of the large size of  $M$ , none of the ineligible items in  $V_0$  is ever selected unless necessary to guarantee model feasibility. When this appears to be necessary though, a minimum number of them is selected and the adaptive update of the eligibility probabilities in (21) automatically compensates for their selection during the next examinees.

Further refinements of the method have been presented in van der Linden and Choi 2020 whereas Choi and Lim (2021) should be consulted for an empirical demonstration of its effectiveness. The current brief review of the method is only to highlight its nature as a separate simultaneous process of data collection with continuous updating of its two item parameters. The process is managed through a simple adjustment of the objective function presented in (22), which now simultaneously serves the two objectives of maximum information in the items about the ability parameters and a minimum number of ineligible items to realize the target values for their exposure rates.

Similar adjustments of the shadow-test model allow us to extend an adaptive testing program with continuous field testing of new items. Suppose a set of new items has been added to the item pool with the purpose of field testing them. The advantages of embedding the items in the adaptive test rather than following the traditional approach with the items administered in a set of separate forms are numerous. The most valuable advantage may be the collection of calibration data under operational conditions, with motivated test takers working under the intended degree of speededness for the test. Add to this the possibility to assign the items adaptively to the test takers, a feature expected to result in a decrease of the calibration sample size comparable to the reduction of test length obtained

when transitioning from a fixed form to an adaptive version of a test. A further increase in efficiency is realized by monitoring the behavior of the items in real time retracting them from the calibration process as soon as any aberrances are observed. (In the traditional approach, the full sample is used before any checks on such aberrances can be conducted.) Another major benefit is automatic linking of the item parameters to the scale already in use for the adaptive test, an advantage due to the cycling of the Gibbs sampler along the ability parameter of each test taker who saw the item during the update of its parameters. Also, unlike fixed-form calibration, the exposure of the field-test items is already controlled during field testing and since they are fully embedded in the content of the adaptive test, it is no longer possible for the test takers to identify their different status.

The setup of choice is to randomly reserve a few positions for the assignment of field-test items near the end of the test prior to each administration of the test, for instance, three positions randomly picked from the last five. Statistically, the best positions would be the very last because they allow for maximal profit from the posterior distribution of the ability parameter. But if this fixed choice would be discovered, less motivation and attempts to memorize the items could follow. Another key part of the setup is permanent presence of the required number of field-test items as a subset in the shadow tests. The presence is necessary to keep the selection of the full test feasible with respect to the complete constraint set in the model.

We use  $x_f$ ,  $f = 1, \dots, F$ , as variables for the set of field-test items in the pool and  $n_o$  and  $n_f$  to denote the required number of operational items and field-test items in the adaptive test. The core of the adjusted model for the selection of the  $k$ th shadow test is

$$\text{maximize } \sum_{i=1}^I \left( S^{-1} \sum_{s=1}^S I(\theta^{(s)}; \xi_i^{(s)}) \right) x_i + \sum_{f=1}^F \left( S^{-1} \sum_{s=1}^S D(\xi_f^{(s)}; \theta^{(s)}) \right) x_f \quad (23)$$

subject to

$$\sum_{i=1}^I x_i = n_o; \quad (24)$$

$$\sum_{f=1}^F x_f = n_f; \quad (25)$$

$$\sum_{i,f \in R_{k-1}} = k - 1, \quad (26)$$

where  $R_{k-1}$  is the joint set of the operational and field-test items already administered and

$$D_s(\xi_f) \equiv S^{-1} \sum_{s=1}^S \left[ \det \left( \text{Cov}^{-1}(\xi_f) + I(\xi_f^{(s)}; \theta^{(s)}) \right) - \det(\text{Cov}^{-1}(\xi_f)) \right] \quad (27)$$

is the posterior expected increase of the criterion of  $D$ -optimality for the case of multiple intentional parameters and a single nuisance parameter (van der Linden 2018a, b, sect. 10.4.1). If the  $k$ th position in the test is for an operational item, the best free operational item in the shadow test according the first criterion in (23) is administered. If it has been reserved for a field-test item, it is the best free item according to the second criterion.

A feature of the adjusted shadow-test model deserving special attention is complete separability of the two problems of optimal selection of the operational and field-test items. The separability is due to the fact that both the two terms of the objective function in (23) and the constraints in (24)–(26) are formulated across separate subsets of items in the pool. The feature is easily maintained by formulating the content constraints to be imposed on the test separately for both types of items. It is also maintained when the penalty term in (22) is added to the objective function. Thanks to the feature, optimal selection of the field-test items does not restrict optimization of the selection of the operational items in any way, or vice versa.

The efficiency of the model was demonstrated for an adaptive test of 25 items from a pool of 250 items for a real-world adaptive test by van der Linden and Jiang (2020). Each simulated administration contained three field-test items among the last five items in the test. In addition to (23)–(26), the shadow-test model had the usual constraints on the content of the items for the test but did not include the earlier option of item-exposure control. After 250 test takers, the final estimates of the difficulty and guessing parameters of the field-test items already showed statistical accuracy fully comparable to what is seen for the typical sample sizes in fixed-form operational testing. For the discrimination parameters, an increase to 1,000 test takers was necessary to reach the same comparability. The setup of the simulation study was written in *R*, the Gibbs sampler in *Java*, and the shadow tests were assembled making calls to a commercial cloud-hosted MIP solver. The average running times was 12 msec for the sampler to update the ability parameters, 25 msec for the update of the field-test parameters, and 53 msec for the calls to the solver.

## 7 Conclusion

The mathematical problem underlying adaptive testing is a constrained discrete optimization problem with a solution that has to be found sequentially without any backtracking to earlier items. The shadow-test approach to adaptive testing was introduced to resolve the problem. Though counterintuitive, it guarantees optimal selection of each of the items without violation of any of the constraints for any of the test takers. Also, rather than slowing down the delivery of the test items, it can be implemented to select the items without any noticeable latency. Later developments include its generalization to real-time sequential test assembly with any testing format for any combination of objectives and set of content specifications. It can

also be used to manage simultaneous testing processes adaptively without sacrificing the efficiency of any of them.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

- Berkelaar M et al. (2021) Interface to Ip\_solve v. 5.5 to solve linear/integer programs. <https://CRAN.R-project.org/package=lpSolve>
- Bixby RE (2012) A brief history of linear and mixed-integer programming computation. *Doc Math Extra Vol IMPS* 2011:107–121
- Chang H-H, van der Linden WJ (2003) Optimal stratification of item pools in alpha-stratified adaptive testing. *Appl Psychol Meas* 27:262–274
- Chen D-S, Batson RG, Dang Y (2010) *Applied integer programming: modeling and solution*. Wiley, New York
- Choi SW, Lim S (2021) Adaptive test assembly with a mix of set-based and discrete items. *Behaviormetrika* 48 (this issue)
- Choi SW, Lim S, van der Linden WJ (2021) TestDesign: an optimal test design approach to constructing fixed and adaptive tests in R. *Behaviormetrika* 48 (this issue)
- Choi SW, Moellering K, Li J, van der Linden WJ (2016) Optimal reassembly of shadow tests in CAT. *Appl Psychol Meas* 40:469–485
- FICO (2021). *XPress Optimization help*. <https://www.fico.com/fico-xpress-optimization/docs/latest/overview.html>
- GNU Linear Programming Kit (2021). *Introduction to GLPK*. URL: <https://www.gnu.org/software/glpk/>
- Geerlings H, van der Linden WJ, Glas CAW (2013) Optimal test design with rule-based item generation. *Appl Psychol Meas* 37:140–161
- Gurobi Optimization LLC (2021). *Gurobi optimizer reference manual*. <https://www.gurobi.com/documentation/9.1/refman/index.html>
- IBM (2021) CPLEX Optimization Studio. <https://www.ibm.com/analytics/cplex-optimizer>
- Koch T et al (2011) MIPLIB 2010. *Math Program Comput* 3:103–163
- Symptom JB, Hetter RD (1985) Controlling item-exposure rates in computerized adaptive testing. *Proceedings of 27th Annual Meeting of the Military Testing Association*. Navy Personnel and Research and Development Center, San Diego, CA, pp 973–977
- Theunissen TJJM (1985) Binary programming and test design. *Psychometrika* 50:411–420
- van der Linden WJ (2005) *Linear models for optimal test design*. Springer, New York
- van der Linden WJ (2006) Equating scores from adaptive to linear tests. *Appl Psychol Meas* 30:493–508
- van der Linden WJ (2010) Sequencing an adaptive test battery. In: van der Linden WJ, Glas CAW (eds) *Elements of adaptive testing*. Springer, New York, pp 103–119
- van der Linden WJ (2016) Lognormal response-time model. In: van der Linden WJ (ed) *Handbook of item response theory: models*, vol 1. Chapman & Hall/CRC, Boca Raton, FL, pp 261–282
- van der Linden WJ (2018a) Optimal test design. In: van der Linden WJ (ed) *Handbook of item response theory: applications*, vol 3. Chapman & Hall/CRC, Boca Raton, FL, pp 167–195
- van der Linden WJ (2018b) Adaptive testing. In: van der Linden WJ (ed) *Handbook of item response theory: applications*, vol 3. Chapman & Hall/CRC, Boca Raton, FL, pp 197–227

- van der Linden WJ, Ariel A, Veldkamp BP (2006) Assembling a CAT item pool as a set of linear test forms. *J Educ Behav Stat* 31:81–100
- van der Linden WJ, Chang H-H (2003) Implementing content constraints in alpha-stratified adaptive testing using a shadow test approach. *Appl Psychol Meas* 27:107–120
- van der Linden WJ, Choi SW (2020) Improving item-exposure control in adaptive testing. *J Educ Meas* 57:405–422
- van der Linden WJ, Diao Q (2014) Using a universal shadow-test assembler with multistage testing. In: Yan D, von Davier AA, Lewis C (eds) *Computerized multistage testing: theory and applications*. Chapman & Hall/CRC, Boca Raton, FL, pp 101–118
- van der Linden WJ, Jiang B (2020) A shadow-test approach to adaptive item calibration. *Psychometrika* 85:301–321
- van der Linden WJ, Li J (2016) Comment on Three-element item-selection procedures for multiple forms assembly: an item matching approach. *Appl Psychol Meas* 40:641–649
- van der Linden WJ, Reese LM (1998) A model for optimal constrained adaptive testing. *Appl Psychol Meas* 22:259–270
- van der Linden WJ, Ren H (2020) A fast and simple algorithm for adaptive testing. *J Educ Behav Stat* 45:58–85
- van der Linden WJ, Veldkamp BP (2004) Constraining item exposure in computerized adaptive testing with shadow tests. *J Educ Behav Stat* 29:273–291
- van der Linden WJ, Xiong X (2013) Speededness and adaptive testing. *J Educ Behav Stat* 38:418–438
- Veldkamp BP (1999) Multi-objective test assembly problems. *J Educ Meas* 36:373–389
- Veldkamp BP, van der Linden WJ (2008) A multiple-shadow-test approach to Symptom–Hetter item-exposure control in adaptive testing. *Int J Test* 8:272–289
- Williams HP (1999) *Model building in mathematical programming*, 4th edn. Wiley, New York

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.