GRADUATE MANAGEMENT ADMISSION COUNCIL

Item Pool Construction Using Mixed Integer Quadratic Programming (MIQP)

Kyung T. Han and Lawrence M. Rudner

GMAC® Research Reports • RR-14-01 • June 10, 2014

Abstract

This study uses mixed integer quadratic programming (MIQP) to construct multiple highly equivalent item pools simultaneously, and compares the results from mixed integer programming (MIP). Three different MIP/MIQP models were implemented and evaluated using real CAT item pool data with 23 different content areas and a goal of equal information functions across pools and within each content area. The study addresses two important practical questions: (a) how many evaluation points should the objective functions of the MIP/MIQP models use when the targets have numerous non~$N(0,1)$ distributions, and (b) how should the solver be structured when an item bank is gigantic? The study finds that all three MIP/MIQP models could be used effectively to construct highly parallel item pools and content bins when five evaluation points were used. Utilization of these techniques can replace current laborious manual pool construction methods.

Introduction

For long-term quality control of computerized adaptive testing (CAT) programs, it is crucial to construct and maintain quality item pools that are consistent over time in terms of their psychometric properties and their match to the ability distributions of the test takers. Regardless of the adaptive algorithm, consistency in pool quality is a necessary condition for consistency in the score accuracy for test takers.

Construction of multiple parallel item pools is often challenging, however, because of the number of factors to be considered (e.g., bank information, content balancing, exposure rate, response time, etc.) and the limited number of items available in the item bank. In applied settings, the goal is often to maintain consistency in pool information functions across pools and to balance content in terms of the number of items within each content area in each pool. Item pool construction is usually performed manually using sampling techniques. Constructing multiple parallel item pools that meet all the pool specifications by hand, however, is very labor intensive, especially when there are numerous content constraints and the number of item pools to be constructed and/or the number of items for each pool is large.

This study investigates the feasibility of using mixed integer programming (MIP) and mixed integer quadratic programming (MIQP) to construct multiple highly equivalent item pools that meet content, exposure, and psychometric constraints, including the goal that each content area should have the equivalent information functions across pools. The study explores three models using three different evaluation points in objective functions. The quality of the approaches is evaluated in terms of pool information function consistency and performance of the solver under each condition.

Mixed Integer Programming Models

Various industries, from delivery services to financial institutions, have made wide use of *linear programming* (LP) models to optimize resources while maximizing outcomes. In the 1980s, the educational measurement field began adopting LP models for optimal test design for applications of automated test assembly (ATA; van der Linden, 2005). For ATA, the most common approach to LP is to introduce as many 0–1 binary variables as there are items in the bank, and then to let the solver software identify an optimal test design by finding the best combination of binary variables that will yield the maximum (or minimum depending on a problem) objective value (Theunissen, 1985). When multiple test forms are constructed at the same time, the LP often takes the form of *network-flow programming*, in which an array of integer variables ($i \times j$) is determined in order to optimize the flows between i supply nodes and j demand nodes (Armstrong, Jones, & Wang, 1995; van der Linden, 1998). Because the decision variables of this LP model are integers, it is generally known as a mixed integer programming (MIP) model, and will be referred to as such in this paper.

Item pool construction, the problem addressed in this paper, is not technically very different from an automated test assembly. Just as MIP models for ATA can be used to assemble the best sets of tests out of a given item pool, the MIP model can, in theory, also systematically assemble optimal sets of item pools. Ariel, Veldkamp, and van der Linden (2004) used the MIP model to optimally divide an item bank into multiple operational pools with similar content distributions. Van der Linden, Ariel, and Veldkamp (2006) discussed the formation of pools to meet the ability distributions of the targeted test takers while meeting content constraints. In practice, meeting content constraints, often a difficult task in itself, does not necessarily mean that the collection of test questions within a content area will meet the important goal of having similar information functions across pools.

For CAT pool construction, whether done by hand or computer, it is more common to minimize the difference between an actually constructed item pool and a target, which often is determined by the examinee's proficiency distribution. This can be modeled by evaluating the information function conditioned on θ:

$$\text{Minimize} \int_{\theta=-\infty}^{\infty} [\sum_{j=1}^{J} I(j \mid \theta) x_j - \tau_\theta] d\theta \qquad (1)$$

subject to

$$\sum_{j=1}^{J} x_j = n_p, \text{ for each } p \text{ (item pool size for } p) \qquad (2)$$

$$\int_{\theta=-\infty}^{\infty} \sum_{j=1}^{J} (I(\theta) x_j - \tau_\theta)\, d\theta \geq 0, \text{ (lower bound of the objective function)} \qquad (3)$$

where $I(j \mid \theta)$ is the item information for item j at θ, x_j is a 0–1 variable representing the exclusion or inclusion of item j, and τ_θ is a target pool information function value at θ.

When multiple item pools are constructed at once, the MIP model above can be modified as follows by summing across the P pools in the objective function and adding an additional constraint:

$$\text{Minimize} \sum_{p=1}^{P} \int_{\theta=-\infty}^{\infty} [\sum_{j=1}^{J} I(j \mid \theta) x_{pj} - \tau_\theta] d\theta \qquad (4)$$

subject to

$$\sum_{j=1}^{J} x_{pj} = n_p, \text{ for each } p \text{ (item pool size)} \qquad (5)$$

$$\int_{\theta=-\infty}^{\infty} \sum_{j=1}^{J} (I(\theta) x_{pj} - \tau_\theta)\, d\theta \geq 0, \text{ for each } p \text{ (lower bound of the objective function)} \qquad (6)$$

$$\sum_{p=1}^{P} x_{pj} \leq m \text{ for each } j, \text{ (item usage across pools)} \qquad (7)$$

where $p = 1, 2, 3, \ldots, P$ with P being the number of item pools to be constructed at the same time and m being the maximum usage for each item across pools.

Since the minimization of the objective function is limited by the constraint (Equation 6), this model will be referred to in this paper as the *single bound model*

(SBM). SBM is mathematically simple and very straightforward; there are many LP/MIP solvers that can handle such a model. The downside of SBM is that there is a fairly high chance of encountering infeasibility issues during the solving process if τ is not set to a very low value, especially when the values of J and m are small and/or the value of P is large. Readers interested in exploring SBM are referred to Cor, Alves, and Gierl (2009).

If the likelihood of infeasibility issues with SBM appears moderate to severe, a different approach to modeling an MIP is often suggested. Infeasibility issues usually occur because of unrealistic settings for constraints. In practice, however, it is often impossible for practitioners to determine, before they attempt to solve the MIP model, whether constraints are unrealistic given the item bank data. Therefore it is useful to have an objective function that optimizes those constraints that restrict the worst difference between the target and actual pool information functions. This approach is called the *minimax approach* (van der Linden, 2005; van der Linden, & Boekkooi-Timminga, 1989). In the minimax approach, Equations 4 and 6 of the SBM above are replaced with

$$\text{Minimize } \delta \tag{8}$$

subject to

$$\sum_{j=1}^{J} I_j(j \mid \theta) x_{pj} \leq (\tau_\theta + \delta), \text{ for each } p, \text{ for all } \theta \tag{9}$$

$$\sum_{j=1}^{J} I_j(j \mid \theta) x_{pj} \geq (\tau_\theta - \delta), \text{ for each } p, \text{ for all } \theta \tag{10}$$

$$\delta \geq 0. \tag{11}$$

In contrast to the SBM, the MIP model applying the minimax approach allows the actual pool information function to be lower than the target pool information function, which significantly relieves possible infeasibility issues during the solving process. Unlike SBM, the differences between the target and actual pool information functions are controlled by the bands around the target, the width of which is 2δ. In this paper, this model will be referred to as the *minimized band model* (MBM).

There are several issues to be resolved with MBM, however. First, the nature of the MBM, in which the constraints continuously change in each iteration, keeps the solver from effectively reducing the space of possible solutions using mathematical strategies. Another issue with the MBM is the fact that the bandwidth across θ is decided by a single δ. When τ is determined carefully throughout θ and the item bank is of sufficient size and quality, this generally should not pose a problem. If τ is unrealistically specified at a certain point on θ so that the difference between the actual pool information function and τ is unusually large at a given θ level, then it will result in δ which is unnecessarily too large for other θ levels. Therefore, with MBM, finding the minimized δ does not always guarantee the minimized overall difference between the actual pool information function and τ throughout θ.

To overcome the problems associated with SBM and MBM, this study introduces a new objective function. In this approach, Equation 4 of the SBM is replaced with a quadratic term as

$$\text{Minimize} \sum_{p=1}^{P} \int_{\theta=-\infty}^{\infty} [\sum_{j=1}^{J} I(j \mid \theta) x_{pj} - \tau_\theta]^2 d\theta \tag{12}$$

subject to

$$\sum_{j=1}^{J} x_j = n_p, \text{ for each } p \text{ (item pool size for } p) \tag{13}$$

$$\sum_{p=1}^{P} x_{pj} \leq m \text{ for each } j \text{ (item usage across pools).} \tag{14}$$

It should be noted that Equation 6 has been dropped from SBM because Equation 12 already is always above zero as the objective function is no longer linear but quadratic. Switching Equations 4 and 6 with Equation 12 may seem a minor change, but this modification fundamentally changes the optimization in terms of the inherent conceptual and technical definitions. Compared with SBM, the elimination of the lower bound of the objective function (Equation 6) reduces the chance of encountering infeasibility issues. Also, unlike MBM, the difference between the actual pool

information function and τ is always minimized throughout θ. Henceforth, in this paper, this model will be referred to as the *minimized squared difference model* (MSDM). Although MSDM has several advantages over SBM and MBM, it has been used in the field only rarely because most LP solvers cannot handle such quadratic programming models. Only a few of the most advanced solvers recently developed can handle mixed integer quadratic programming (MIQP) problems under very limited conditions.

The conceptual illustrations of SBM, MBM, and MSDM are shown in Figure 1. The reader should note the differences in the possible shapes of the constructed pool information functions.

Purpose of Study

The primary goal of this study is to compare the performance of the two different MIP models (SBM and MBM) and the MIQP model (MSDM) in constructing multiple parallel item pools. Before implementing these MIP/MIQP models, however, two important issues need to be addressed.

The objective functions (Equations 4, 8, and 12) for item pool constructions are based on a continuous scale of θ with integrals. To make the objective function recognizable for the MIP solver and to reduce the intensity of mathematical computation, it is necessary to replace the integrals with summations of objective values across a few discrete evaluation points (i.e., quadrature points). For example, the objective function for SBM (Equation 4) is changed to

$$\text{Minimize} \quad \sum_{p=1}^{P}\sum_{t=1}^{T}[\sum_{j=1}^{J} I(j \mid \theta_t)x_{pj} - \tau_t] \qquad (15)$$

where $t = 1, 2, 3, \ldots, T$ with T being the number of evaluation points (EP) on the θ scale, θ_t being the θ value at evaluation point t, and τ_t the target pool information function at θ_t. The same applies to MBM and MSDM. The first issue then is determining what the optimal number and locations of evaluation points are on the θ scale. Too few evaluation points would prevent tight control of the actual pool information function whereas too many evaluation points may unnecessarily and dramatically increase the solver processing time. Van der Linden (2005, p. 106) suggests that with an $N(0,1)$ expected θ distribution, three or four evaluation points specified at -1.0, 0.0, +1.0 or -1.5, 0.5, 0.5, 1.5 will yield excellent results for a typical ATA. Practically speaking, however, for item pool construction, thetas do not remain $N(0,1)$ and target difficulty is not $N(0,1)$. While one could laboriously tailor the evaluation points for each pool, this study aims to determine a reasonable set of locations of evaluation points that can be universally applied for target information functions that may not closely follow $N(0,1)$.

The second critical part of the solving process is managing the size of MIP/MIQP problems to keep them under the usable computer resource limit. Unlike typical LP problems, MIP problems with 0–1 variables (for example, x in Equation 15) rely heavily on the branch-and-bound (BnB) algorithm (Land & Doig, 1960) to implement the iterative tree search. The size of a solution space is defined by the number of possible combinations of the 0–1 variables. For example, if an ATA problem was to assemble a test form consisting of 30 items selected from an item pool of 500 items, the size of the solution space for the problem would be

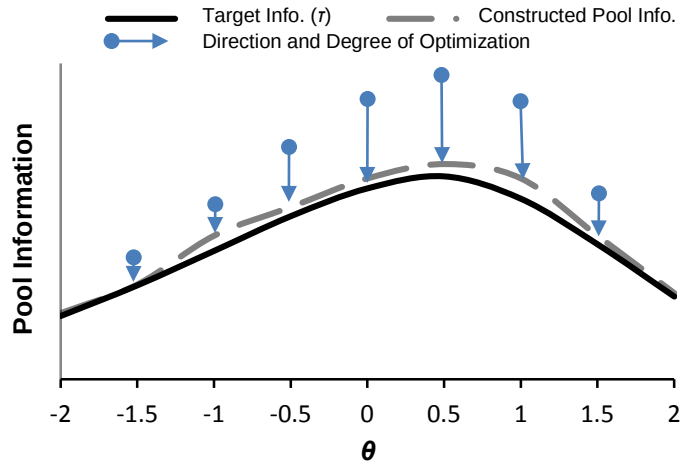
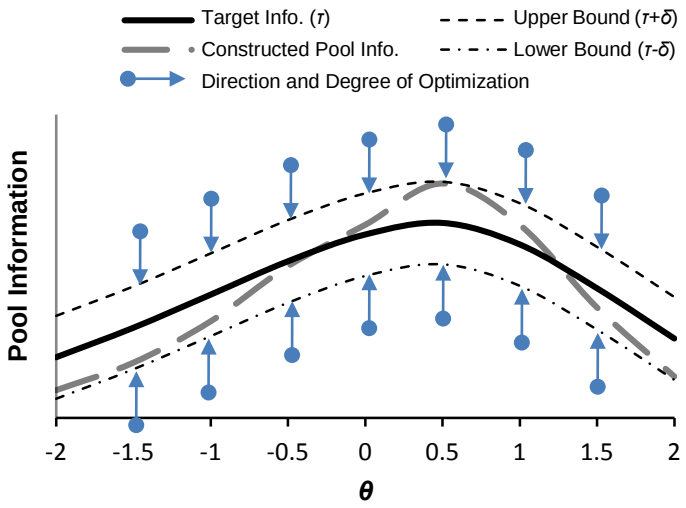
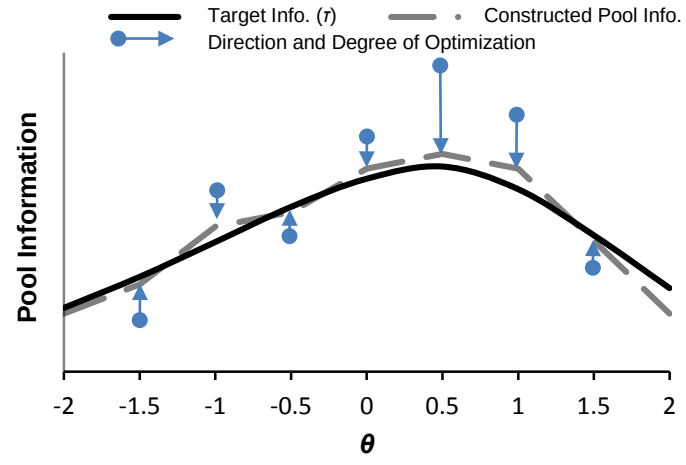
$$\binom{500}{30} = \frac{500!}{30!(500-30)!} \cong 1.445e^{48}. \qquad (16)$$

Although 1.445E + 48 is large, many solvers can effectively reduce the problem size by eliminating infeasible solutions (according to the constraints) and skipping (or cutting) unpromising solutions using various mathematical strategies. With CAT pool construction, however, the size of a solution space can become unmanageable. For example, if 12 parallel item pools (500 items per pool) were constructed from an item bank with 10,000 items, the solution space would be

$$\binom{10000}{500\times 12} = \frac{(10000)!}{6000!(10000-6000)!} \cong 5.794e^{2920}. \qquad (17)$$

Only a handful of advanced, high-performance solvers could theoretically manage the computer resources (memory and storage) needed to handle such an enormous MIP/MIQP problem. Assuming the problem could be solved, it is unknown whether an advanced solver would be able to finish the solving

Figure 1. Three Optimization Models (Excluding Constraints for Item Exposure Control)

Model	Demonstration	Main Part of Model
Model 1: Single Bounded Model (SBM) Type: MIP		Minimize $\sum_{p=1}^{P}\int_{-\infty}^{\infty}[\sum_{j=1}^{J}I(j\mid\theta)x_{pj}-\tau_{\theta}]d\theta$ subject to $\sum_{j=1}^{J}I(\theta)x_{j}-\tau_{\theta}\geq 0$, for each p.
Model 2: Minimized Band Model (MBM) Type: MIP		Minimize δ subject to $\sum_{j=1}^{J}I_{j}(j\mid\theta)x_{pj}\leq\tau_{\theta}+\delta$, and $\sum_{j=1}^{J}I_{j}(j\mid\theta)x_{pj}\geq\tau_{\theta}-\delta$, for each p across $\theta[-\infty,\infty]$.
Model 3: Minimized Squared Difference Model (MSDM) Type: MIQP		Minimize $\sum_{p=1}^{P}\int_{-\infty}^{\infty}[\sum_{j=1}^{J}I(j\mid\theta)x_{pj}-\tau_{\theta}]^{2}d\theta$

process within a realistic time frame. Therefore, in this study, item pool construction is performed in two different ways. In the first method, a whole item bank is modeled using MIP/MIQP using an advanced solver. In the second method, the item bank is divided into subgroups first, followed by implementation of the MIP/MIQP models. The performance of the solver when the problem tree is astronomically large is evaluated by comparing the two methods.

Once these two research questions are addressed, the three MIP/MIQP models are compared and evaluated. A comprehensive discussion of the study findings draws important guidelines for item pool construction using MIP/MIQP techniques.

Method

Pool Specification

For this study, 12,000 quantitative items were randomly selected from the Graduate Management Admission Test® (GMAT®) item bank (i.e. master pool). This study used a scenario[1] in which 12 item pools had to be constructed to meet the following realistic requirements:

1. Each item pool must consist of items from 23 mutually exclusive combinations of content areas, cognitive skills, and application levels (in this paper, the combinations are simply referred to as content area, and the mutually exclusive collections of items within a content area are referred to as bins);
2. The number of items for each content area is equal to a prespecified value, n_{kp}, (k being an index for each content area);
3. An item cannot be included in more than two of the 12 pools; and
4. An item cannot be included in more than 1 of 4 consecutive pools.

The number of available items for each content area (N_{kp}) in the item bank ranged between 233 and 938, and the prespecified number of items for each content area (n_{kp}) per item pool ranged between 27 and 40. The distribution of item difficulties within each of the pools deviates greatly from $N(0,1)$

Models

The four aforementioned pool requirements were added as constraints into SBM, MBM, and MSDM. For the content area component, C_{jk}, a matrix of 0–1 constants that indicates the content area of each item, was added to the models, in which

$$C_{jk} = \begin{cases} 1 & \text{if item } j \text{ belongs to content area } k \\ 0 & \text{otherwise.} \end{cases} \quad (18)$$

For example, the final SBM model could be expressed as,

$$\text{Minimize} \quad \sum_{k=1}^{K}\sum_{p=1}^{P}\sum_{t=1}^{T}[\sum_{j=1}^{J} I(j\,|\,\theta_t)x_{pj}C_{jk} - \tau_{kt}] \quad (19)$$

subject to

$$\sum_{j=1}^{J} x_{pj} = n_p, \text{ for each } p \text{ (pool size)}, \quad (20)$$

$$\sum_{j=1}^{J} x_{pj}C_{jk} = n_{kp}, \text{ for each } k \text{ and } p \text{ (content areas)}, \quad (21)$$

$$\sum_{j=1}^{J} I(j\,|\,\theta_t)x_{pj}C_{jk} - \tau_{kt} \geq 0, \text{ for each } k, p, \text{ and } t \text{ (information targets)}, \quad (22)$$

$$\sum_{p=1}^{P} x_{pj} \leq 2, \text{ for each } j \text{ (maximum item use)}, \quad (23)$$

$$\sum_{p}^{(p+3)\leq P} x_{pj} \leq 1, \text{ for each } j \text{ (consecutive overlap constraint)}. \quad (24)$$

To find the optimal number and location of the evaluation points on the θ scale, three different combinations of number and location were attempted: 1 ($\theta = 0$), 3 ($\theta = -2, 0$, and 2), and 5 evaluation points ($\theta = -2, -1, 0, 1$, and 2). The analysis thus consists of the

[1] This is a hypothetical scenario. The pool specifications and constraints are not those of the operational GMAT exam.

following: $P = 12$ pools to be created, $J = 12{,}000$ items in the bank, and $K = 23$ content areas. We will evaluate $T = 1$, 3, and 5 evaluation points.

As a baseline, 12 pools were constructed manually using sampling techniques. In this manual construction, the mean and standard deviations (SD) of *a*- and *b*-parameters of each pool were matched to other item pools and the maximum information was targeted at about $\theta = 0.85$. The manually constructed item pools were compared with the item pools constructed using the MIP/MIQP optimization.

Implementation

As MIP/MIQP models, SBM, MBM, and MSDM were built using the optimization modeling software, *AIMMS 3.10FR2* 64-bit edition (www.aimms.com) and one of its most advanced solvers, *CPLEX 12.1* (www.cplex.com), which could handle both MIP and MIQP. To ensure that the objective functions under the different evaluation point conditions were comparable, each of the objective functions was divided by the number of evaluation points. The relative optimality tolerance was set to 5% of best LP bounds, for all three models. To expedite the iterative process of the BnB algorithm, the cutoff criteria for SBM, MBM, and MSDM were set to about twice the size of the corresponding absolute optimality tolerances, which were derived from the relative tolerance values, based on the results from multiple preliminary runs. A best-estimate strategy was used in the node selection to start from a node after all infeasible integer solutions were removed. The study also employed strong branching, in which the variable selections were based on partially solving a number of subproblems to see which branch was the most promising, because this approach is known to be very effective on large optimization problems. The solver was set to run in the parallel thread mode using all available CPU cores.

As mentioned earlier, this study employed two different means of implementation. First, the item bank ($N = 12{,}000$) was divided into 23 mutually exclusive content subgroups. Then the solving process was performed for each of the 23 subgroups (i.e., 23 separate runs of the solver). In the second implementation method, the solver was set with the goal of determining all 144,000 binary variables (the $p \times j$ matrix of x's, where $P = 12$ and $J = 12{,}000$) to construct 12 item pools with all 23 content areas at once.

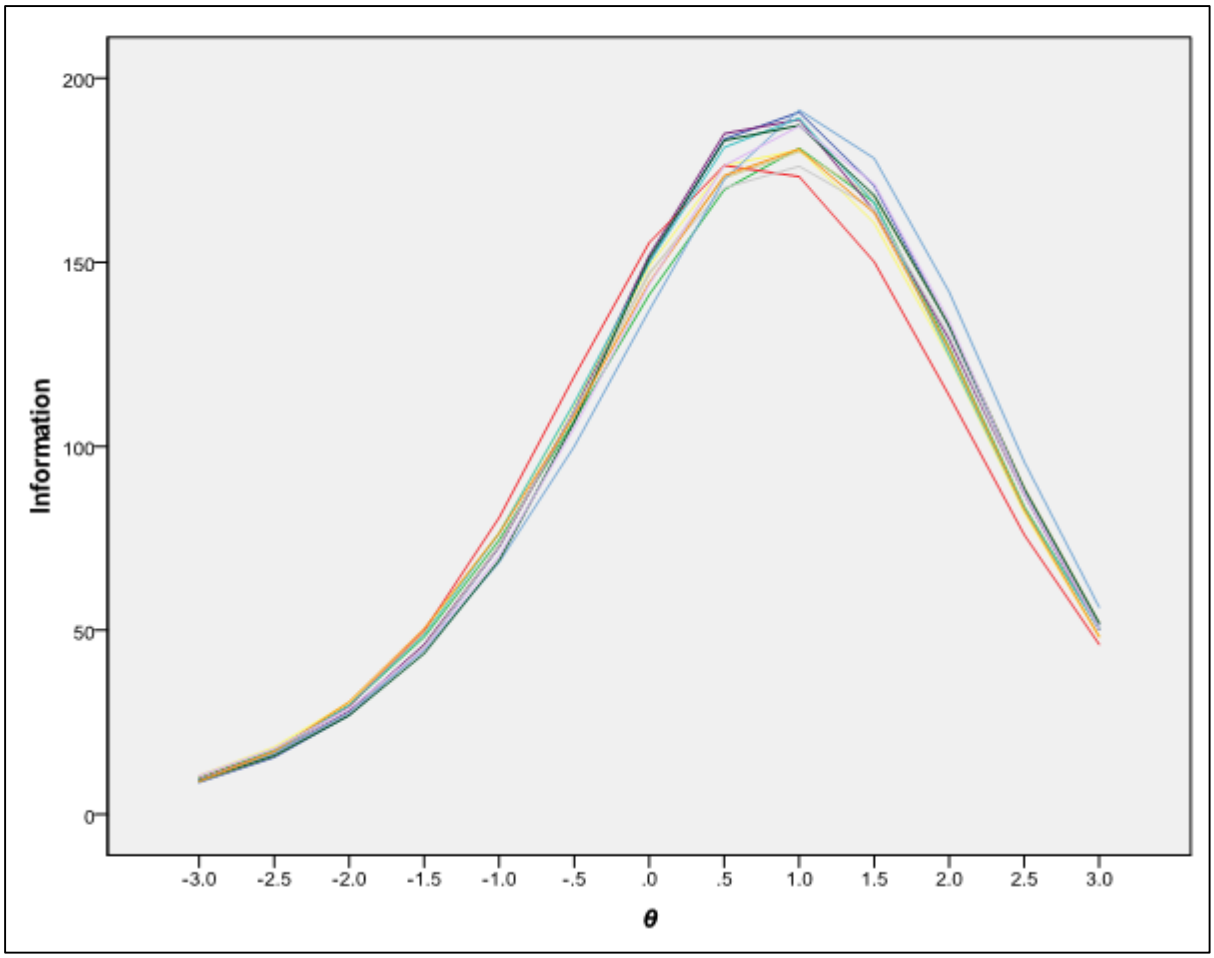
The computer system used for the solving process was a virtual machine built on Microsoft® Windows® Server 2003 64-bit edition with a dedicated Intel® Zeon CPU with four cores running at 2.80 GHz. The computer had 16 GB of physical memory. On a practical note, it is advisable to use a 64-bit operating system in order to avoid system crashes associated with the memory management when the MIP/MIQP problem is very large and parallel thread running is required.

Results

The pool information functions of the 12 manually constructed item pools are shown in Figure 2. Although the pool information functions across the 12 item pools differed slightly in where they peaked, the overall shapes of the pool information functions were quite similar. When we examined the information functions at the item bin level (i.e., subgroups by content area), however, the information functions across the pools within each bin were inconsistent. For example, as shown in Figure 3, the information functions for Bins 6 and 9 differed in their peak points by about 200% between some pools. For Bins 16 and 17, the overall shapes of the bin information functions varied greatly from one pool to another. In sum, manual item pool construction using the sampling technique matching the mean and standard deviation of item parameter values across item pools results in an acceptable level of consistency in pool information function at the pool level. At the lower level (i.e., item bin or item content area), however, the item bin quality might be inconsistent across item pools (or over time). Item pools constructed manually in this manner have been acceptable to test developers performing real-world applications. It is undeniable, however, that the inconsistency in the psychometric properties among the item pools when the pools were manually constructed posed a far from ideal result, leaving much room for improvement.

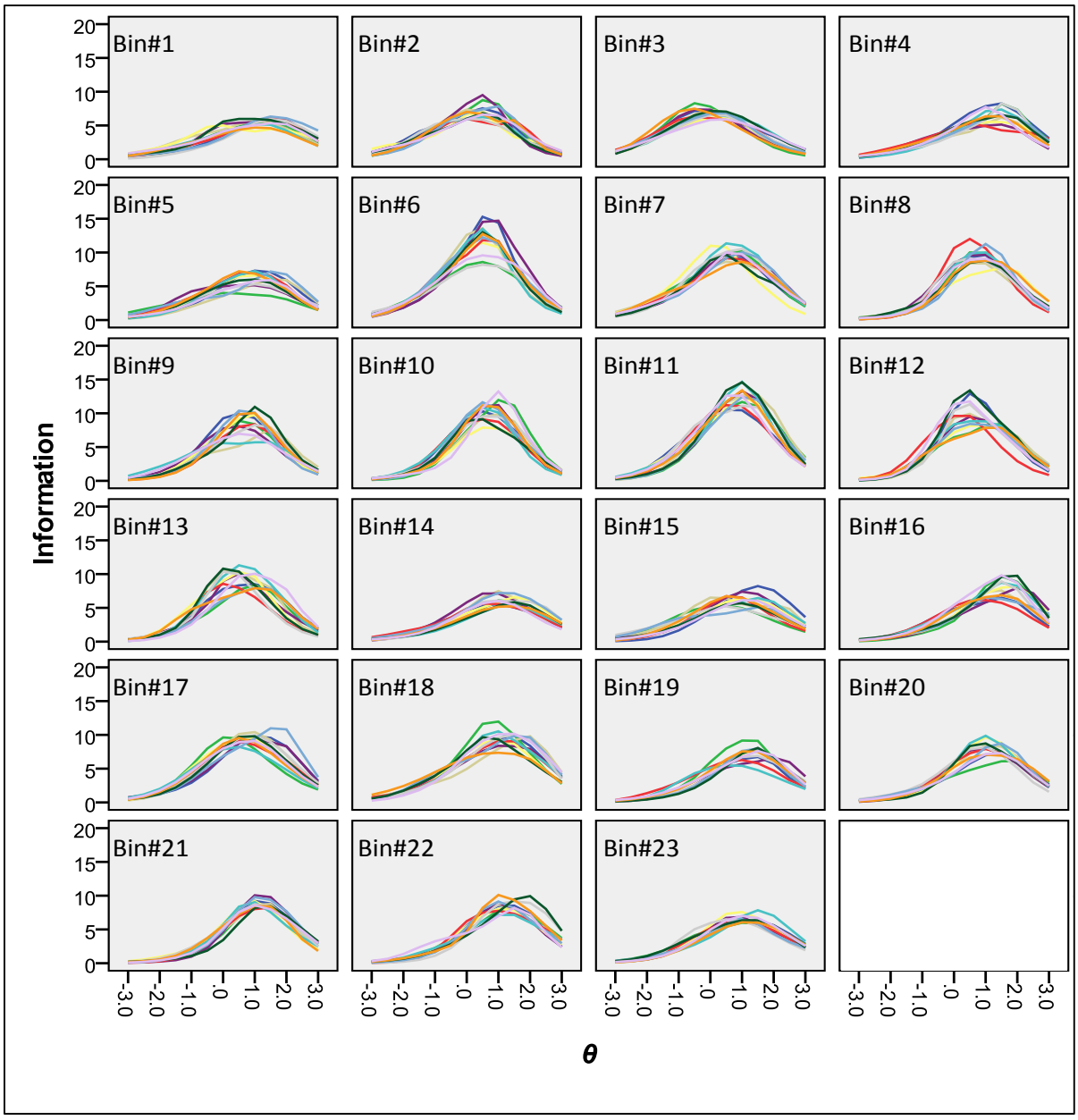
The item pool constructions using the MIP/MIQP optimization-based SBM, MBM, and MSDM were conducted next and the information functions of these item pools at the item pool level are shown in Figure 4.

Figure 2. Pool Information Functions of 12 Manually Constructed Item Pools



When the objective functions were controlled at one evaluation point (1EP; $\theta = 0$), the pool information functions of the item pools emulated those seen in the manual construction cases in Figure 3. In addition, the pool information functions of most of the item pools in the 1EP condition tended to exceed the target where $\theta > 1$. When the objective functions were controlled at three evaluation points with two standard deviation intervals (3EP; θ = -2, 0, and 2), the MBM model yielded the most consistent pool information functions across the 12 pools. The pool information functions in the 3EP condition tended to be closer to the target than those in the one-evaluation point condition; however, with SBM and MSDM, many pools showed pool information functions lower than the target where the pool information functions peaked around $0 < \theta < 2$. Also, it should be noted that the information functions were not tightly controlled at the middle of the distribution (e.g., -1 and 1). When the objective functions were controlled at five evaluation points (5EP; θ = -2, -1, 0, 1, and 2), pool information functions of the 12 pools were on top of each other as well as right on the target throughout θ with all three MIP/MIQP models. This represents a dramatic improvement over the manual pool construction (Figure 1) in terms of quality control of item pool construction. Based on the results of the one-, three-, and five-evaluation point conditions in this study, it is reasonable to conclude, as practical suggestions that: (a) the objective functions should be evaluated where the pool information functions are expected to peak, (b) the

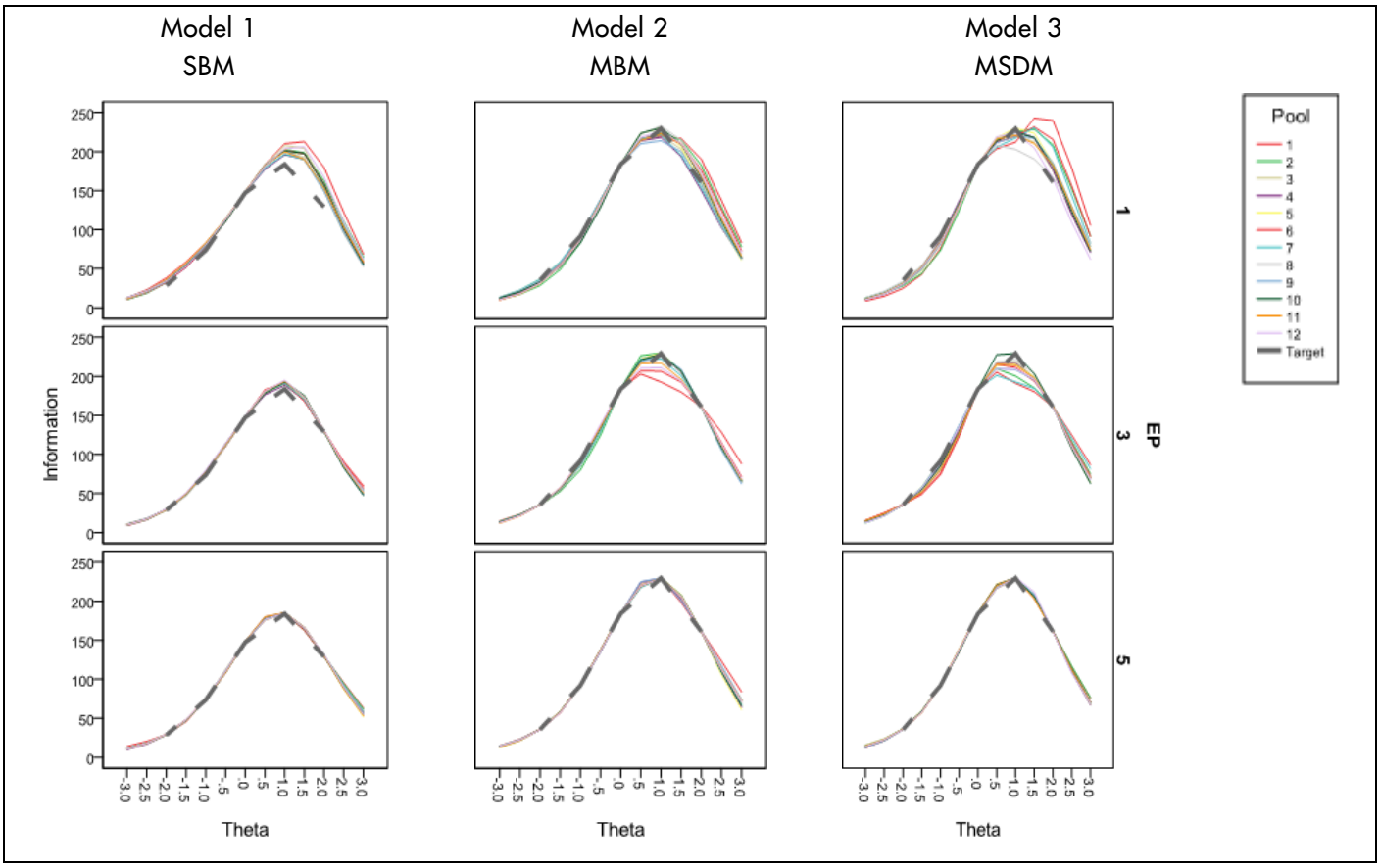
Figure 3. Bin Information Functions of 12 Manually Constructed Item Pools



interval of evaluation points should be one standard deviation or less, and (c) there should be at least five evaluation points to control the objective functions effectively throughout a wide range of θ.

Although the results show that the information functions of the 12 pools were controlled effectively at the pool level with the five evaluation points, it is still crucial to examine the consistency of the 12 item pools at the critical item bin level because, as seen in Figure 3, consistency in pool information does not necessarily translate to consistency in information functions at the content or bin level, even if the items are drawn from these mutually exclusive bins. Figures 5, 6, and 7 show the bin information functions of the item pools

Figure 4. Information Functions of 12 Pools Constructed by Three Models at the Pool Level



constructed by SBM, MBM, and MSDM, respectively.[2] With three or fewer evaluation points, bin information functions often differ substantially from the target as well among the 12 pools. With five evaluation points, bin information functions were right on target across the 12 pools within each content area. In a few cases, the 12 pools differed from each other within each bin when $\theta > 2$, but the differences observed were negligible. Overall observations on the bin information functions led to the conclusion that evaluation of the objective functions on the five evaluation points effectively controlled the information at the item bin level as well as at the item pool level. In this study, there were no meaningful differences among any of the MIP/MIQP models in terms of quality of the constructed item pools. The carefully chosen targets (τ) were well supported by the large item bank and the overlap constraints were achievable. In terms of the time required to perform the solving processes, however, there were substantial differences among SBM, MBM, and MSDM as shown in Table 1. When the objective functions were evaluated at one evaluation point (1EP), the solver with the MBM objective performed the fastest. The average processing time across 23 runs (for 23 item bins, with 12 pools for each bin) was nine seconds. In the 3EP condition with the SBM and MBM, processing time increased compared with the 1EP condition, but the MBM still resulted in the shortest processing time.

[2] Results only for Bins 6, 9, 16, and 17 were reported due to space limitations, but results for other bins are available from the first author upon request.

Table 1. Processing Time (in Seconds) and Final Objective Value for Each MIP/MIQP Model			
Model	Number of Evaluation Points	Time	Objective Value
SBM	1	57 (37.75)	0.478 (0.07)
	3	639 (461.49)	0.491 (0.04)
	5	921 (629.03)	0.510 (0.03)
MBM	1	9 (6.72)	0.037 (0.01)
	3	80 (67.92)	0.042 (0.00)
	5	2868 (3433.47)	0.044 (0.00)
MSDM	1	791 (540.09)	0.018 (0.00)
	3	487 (311.61)	0.017 (0.01)
	5	1106 (726.62)	0.021 (0.00)
Note: Two stop criteria were used for the solving process: (a) CPU time (10,000 sec) or (b) objective value with relative tolerance.			

Under the 5EP condition, which resulted in near optimal pools and bins, the processing time for MBM jumped to an average of 2,868 seconds (47.8 minutes), and 3 out of the total of 23 runs could not finish within the time limit (10,000 seconds or 166.6 minutes) even though the objective function values of those three runs were close to absolute optimality tolerance. For the 5EP condition, the SBM model took the shortest length of time to solve, with an average of 921 seconds or approximately 16 minutes.

MSDM took the longest time to solve when there was only one evaluation point, but interestingly, compared with the other models, MSDM processing time did not necessarily increase as the number of evaluation points increased. The tendency for processing times to be influenced by the number of evaluation points and choice of MIP/MIQP models does not lend itself directly to generalization; however, overall processing times reported were informative enough to help choose the best MIP/MIQP models based on the number of evaluation points considered. In fact, regarding the person-hours typically required for manual construction, the processing time with any of the three MIP/MIQP models was extremely fast and the differences among the models in term of processing time were of no practical significance.

A simultaneous solving, where all 23 bins (each with 12 pools) were constructed at the same time, was also attempted to determine whether the solver could still solve the different MIP/MIQP models effectively given an extremely large problem size. With SBM and MSDM, the solver falsely concluded that there was no feasible solution after the presolving process. When MBM was used, the solver ran until the solving process was forcedly terminated when it ran out of computer resource after 1,219,846 seconds (more than two weeks) of running (1,964,920 nodes explored). The minimization objective function value of 13.820 was the best solution found after two weeks. This is far greater than the sum of the objective values from the 23 separate runs (0.904) in the 5EP condition.

Discussion and Conclusion

This study addressed three important research goals related to item pool construction using MIP/MIQP techniques. The first goal concerned identifying how many evaluation points were needed to control information functions effectively at both the item pool level and the item bin (i.e., content area) level regardless of the distributions of information within each bin. In the studied condition, where the information function between -2 and 2 of the θ scale was of the most interest, at least five evaluation points were required to control the information function effectively across all the bins. The evaluation points did not have to be equidistant, but study findings suggested that the intervals between evaluation points should be one standard deviation or less. If the information functions need to be controlled

Figure 5. Bin Information Functions for 12 Pools Constructed With SBM (MIP)

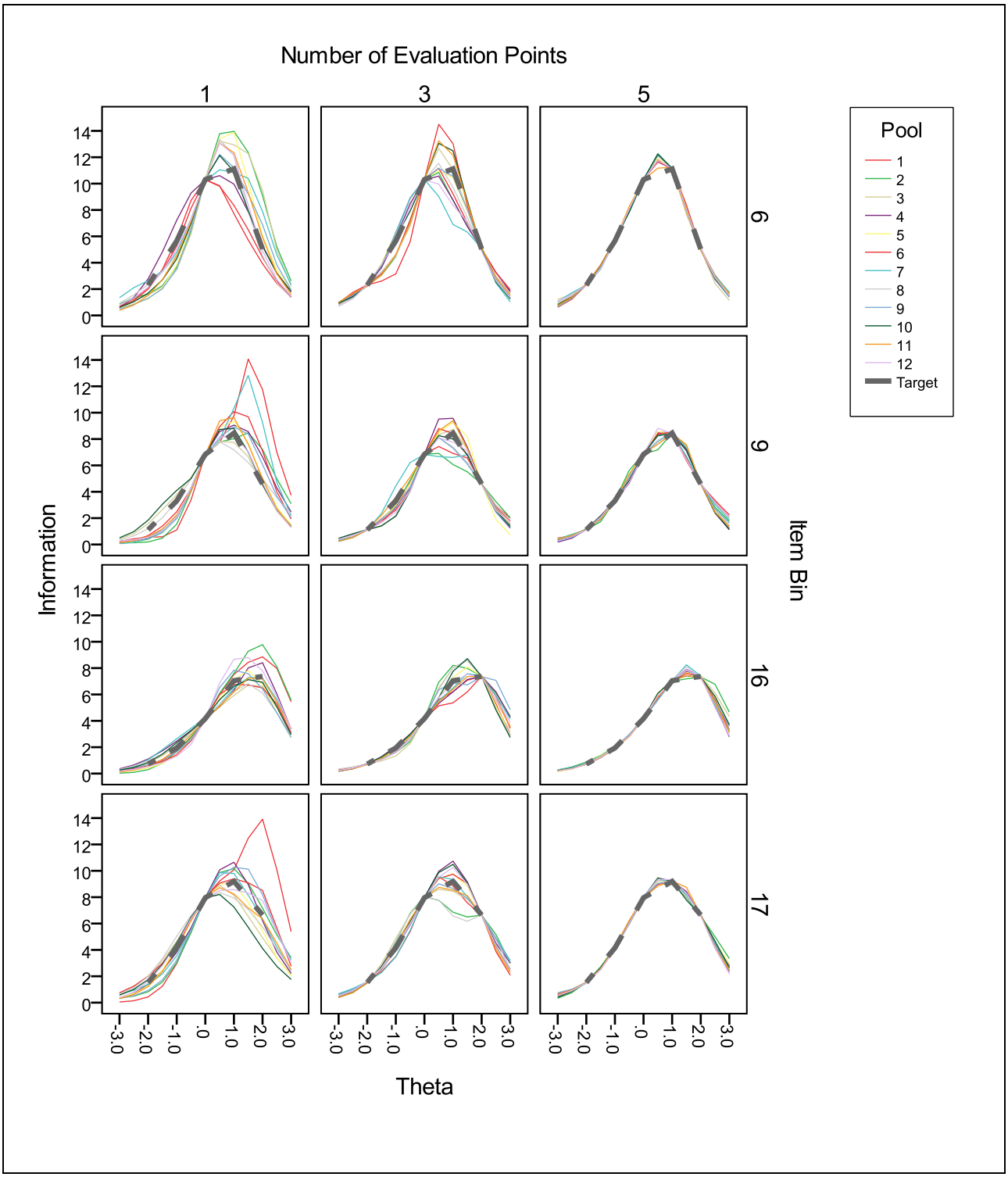


Figure 6. Bin Information Functions for 12 Pools Constructed With MBM (MIP)

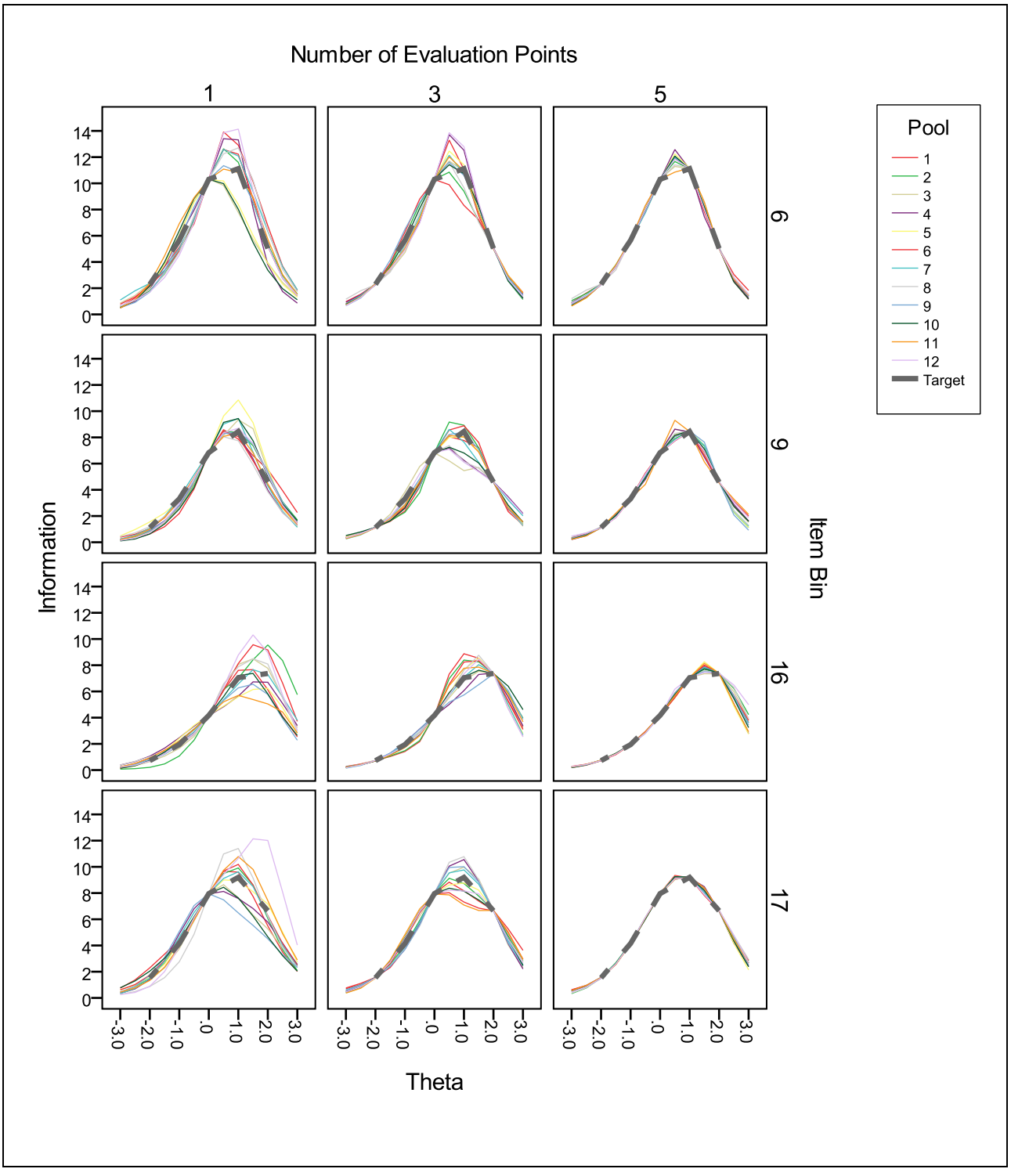
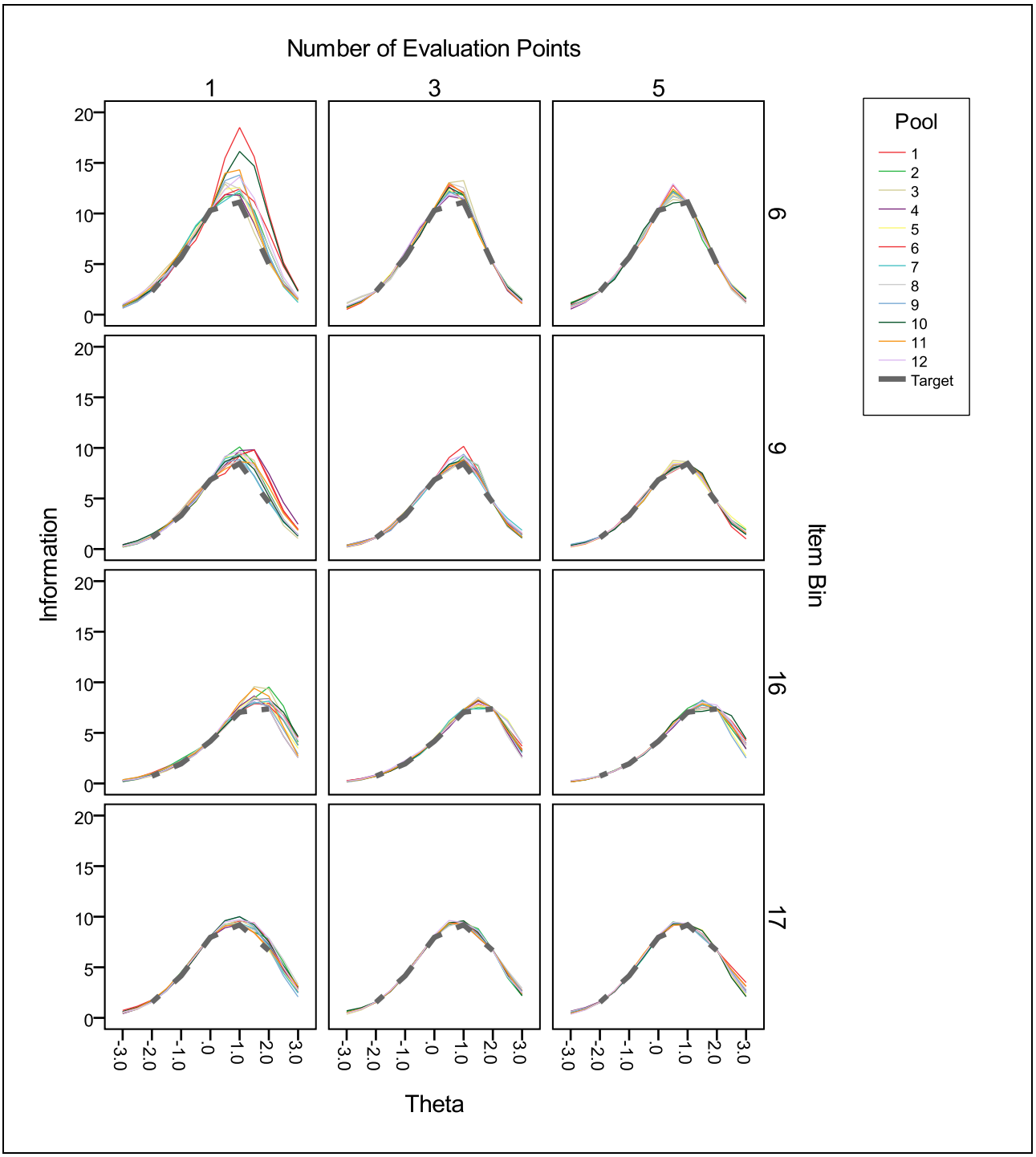


Figure 7. Bin Information Functions for 12 Pools Constructed With MSDM (MIQP)



for a wider range on the θ scale, then more evaluation points would be required. One could tailor evaluation points to each content area and possibly use fewer evaluation points. Setting evaluation points of -2, -1, 0, 1, and 2 worked well, however, regardless of target shapes.

This study's second research goal was to determine whether one of the most recently developed solvers could handle the construction of large multiple item pools simultaneously. Under the studied condition, where 12 item pools were constructed from an item bank of 12,000 items, some of the solvers could not even reach the actual solving stage. Even when the solving process was successfully initiated, few optimal solutions were found within a realistic processing time setting; yielding an unsatisfactory outcome. On the other hand, when the item bank was broken into smaller mutually exclusive subgroups (each with fewer than 1,000 available items to choose from), the solvers were able to find near-optimal solutions for constructing 12 item pools within an hour, except for a few cases with the MBM. Thus, it is advisable to stratify the item bank and item pools into mutually exclusive, smaller subgroups whenever possible. If stratifying the item bank is not feasible and the item bank is huge, other approaches such as van der Linden's big-shadow-test approach (2005) or sequential solving (constructing one item pool with each run) might be used to reduce the size of the problem tree. Note, however, that such approaches could compromise the level of optimization across the item pools.

The last research objective—and the main concern of this study—was to compare the performances of the SBM, MBM, and MSDM models in item pool construction. When the target was carefully established and the item bank had a capacity to fully support the target, all three MIP/MIQP models proved to be feasible and effective in item pool construction with five evaluation points. It bears repeating, however, that this study involved the use of an item bank with a total of 12,000 available items, a size large enough to avoid infeasibility issues. When fewer items are available, then MSDM might be a more preferable choice over the other MIP models because MSDM usually has fewer constraints than the comparable SBM, meaning it is less likely to encounter infeasibility issues. MSDM seeks the most optimal solution at each of the EP levels, whereas MBM's best solution is not necessarily the most optimal at each EP. For more information about potential differences in performance among the models when an item bank is insufficient for fully supporting a target (i.e., when an item bank is deficient), see Appendix A.

Depending on choice of item selection method, item exposure control, and other adaptive algorithms, item pool quality may not directly influence CAT administration for every single test taker at any given time. Presuming, however, that an entire item pool is managed to be evenly utilized, consistency of item pool quality across the constructed item pools would, in the long run, become one of the most important factors in deciding the quality of the measure itself. Therefore, it cannot be overstated that the dramatic improvement seen in item pool construction with the MIP/MIQP technique also would be a significant step forward in ensuring the test validity.

Contact Information

For questions or comments regarding study findings, methodology or data, please contact the GMAC Research and Development Department at research@gmac.com.

The views and opinions expressed in this article are those of the authors and do not necessarily reflect those of the Graduate Management Admission Council (GMAC).

Acknowledgements

The authors wish to thank Paula Bruggeman, Writer/Editor, Manager, Research and Development, Graduate Management Admission Council (GMAC), for her editorial review.

.

References

Ariel, A., Veldkamp, B. P., & van der Linden, W. J. (2004). Constructing rotating item pools for constrained adaptive testing. *Journal of Educational Measurement, 41*, 345–359.

Armstrong, R. D., Jones, D. H., & Wang, Z. (1995). Network optimization in constrained standardized test construction. In K. D. Lawrence (Ed.), *Applications of management science: Network optimization applications* (Vol. 8, pp. 189–212). Greenwich, CT: JAI Press.

Cor, K., Alves, C., & Gierl, M. (2009). Three applications of automated test assembly within a user-friendly modeling environment. *Practical Assessments, Research & Evaluation, 14*(14), pp. 1–23.

Huitzing, H. A., Veldkamp, B. P., & Verschoor, A. J. (2005). Infeasibility in automated test assembly models: A comparison study of different models. *Journal of Educational Measurement, 42,* 223–243.

Land, A. H., & Doig, A. G. (1960). An automatic method of solving discrete programming problems. *Econometrica, 28*(3), 497–520.

Theunissen, T.J.J.M. (1985). Binary programming and test design. *Psychometrika*, *50*, 411–420.

van der Linden, W. J. (1998). Optimal assembly of psychological and educational tests. *Applied Psychological Measurement*, *22*, 195–211.

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

van der Linden, W. J., Ariel, A., & Veldkamp, B. P. (2006). Assembling a CAT item pool as a set of linear test forms. *Journal of Educational and Behavioral Statistics, 31*, 81–100.

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

Appendix A

Cases With a Deficient Item Bank

The study revealed that the three MIP/MIQP models were effective in constructing item pools when: (a) the objective functions based on target information functions were carefully established, and (b) the item bank had a sufficient number of items available to support feasible solutions. In reality, however, issues of infeasibility arise frequently. As Huizing, Veldkamp, and Verschoor (2005) pointed out, there are two main reasons for infeasibility: a contradiction between the demands and/or a deficient item bank. A contradiction between demands in a mathematical model usually is easy to identify and fix. On the other hand, determining whether or not the item bank is deficient before actually attempting item pool construction can be extremely challenging in practice. For this reason, the SBM, MBM, and MSDM were compared under the condition of a deficient item bank.

Item Bank and Pool Specification

A total of 3,000 items were selected from the GMAT exam item bank for the quantitative section. The item pool consisted of 10 bins defined by five content areas and two cognitive skills. To render the item bank insufficient for meeting all item pool specifications, the information targets for some bins were set intentionally higher than what the item bank actually could support. All other conditions and constraints, including item exposure restrictions, remained the same as in the earlier analysis. SBM, MBM, and MSDM (models) were used to construct 10 parallel item pools, and objective functions were evaluated at five evaluation points (-2, -1, 0, 1, and 2). The solver was set to construct 10 parallel pools for all 10 bins at once. Because indeterminacy (i.e., failing to find a single, absolutely optimal solution)

was likely, a time limit of 10,000 seconds (167 minutes) was established.

Different Behaviors of SBM, MBM, and MSDM

During the presolving process using SBM, the solver discovered there was no feasible solution for item pool construction and stopped running. This came as no surprise since the item bank was set intentionally to be insufficient for satisfying the target information constraint (the lower bound of the SBM seen in Equation 6) for the study. Knowing an item bank is deficient, test developers can consider possible approaches to fix the infeasibility issue, for example, by adding more items, replacing items with newer ones with higher information at problematic proficiency areas, and/or making the target information more realistic. In practice, however, identifying the exact reason for an infeasibility issue is extremely challenging, especially when there are numerous constraints and the item pool structure is complex. The shortcoming of the SBM, therefore, is not only its incapacity to solve but also its inability to produce useful information to fix infeasibility issues when faced with a deficient item bank.

When the MBM was used for item pool construction, the solver, within the time limit, successfully found an acceptable solution in which the difference between the linear optimum and the actual integer solution was less than 5%. The differences in bin information functions among the 10 constructed pools and the target are reported in Figure A.1. As shown in Figure A.1, the constructed item pools for Bins 5, 6, 7, and 8 were nearly parallel and right on target in terms of bin information. For the other bins, however, the bin information functions of the constructed item pools exhibited obvious differences from the targets. Based on the MBM results, it was easy to identify which bins were problematic. Examining each of the problematic bins, however, it was difficult to determine the exact cause of the problem. For Bin 3, for instance, all pools showed information functions that were below the target at -2, whereas all pools well exceeded the target at 0. Also, at evaluation points 1 and 2, there were larger variances in information functions across pools. Thus, while the results from the MBM seemed to be useful for identifying the item bins for which the item bank was deficient, they were not specific enough to pinpoint the problematic areas across evaluation points.

With the MSDM, the results showed that 10 constructed pools were nearly parallel to each other across bins and evaluation points. As displayed in Figure A.2, the 10 item pools showed identical bin information functions at every evaluation point even where the item bank capacity was not satisfying the target information. It is important to note that the differences in bin information function between the constructed pools and the target were minimized at each evaluation point with the MSDM. With Bin 3, for example the bin information functions were substantially off the target at $\theta = 1$ and 2 with the MBM (Figure A.1), while the MSDM resulted in the 10 parallel pools that were right on the target at the same evaluation points. In sum, this example clearly shows the benefits of using the MSDM for item pool construction when an item bank is potentially deficient. First, because the squared differences between a target and constructed item pools are always minimized at every evaluation point, the MSDM effectively constructs multiple parallel pools even if a target is unrealistic. Second, with the MSDM, problematic areas across bins and evaluation points are clearly revealed. Thus, test developers can determine effective changes on the target and/or items in the bank to relieve the item bank deficiency issue without iterative trial-and-error processes.

Figure A.1. Pool Information Functions for 10 Pools Constructed With MBM (MIP)

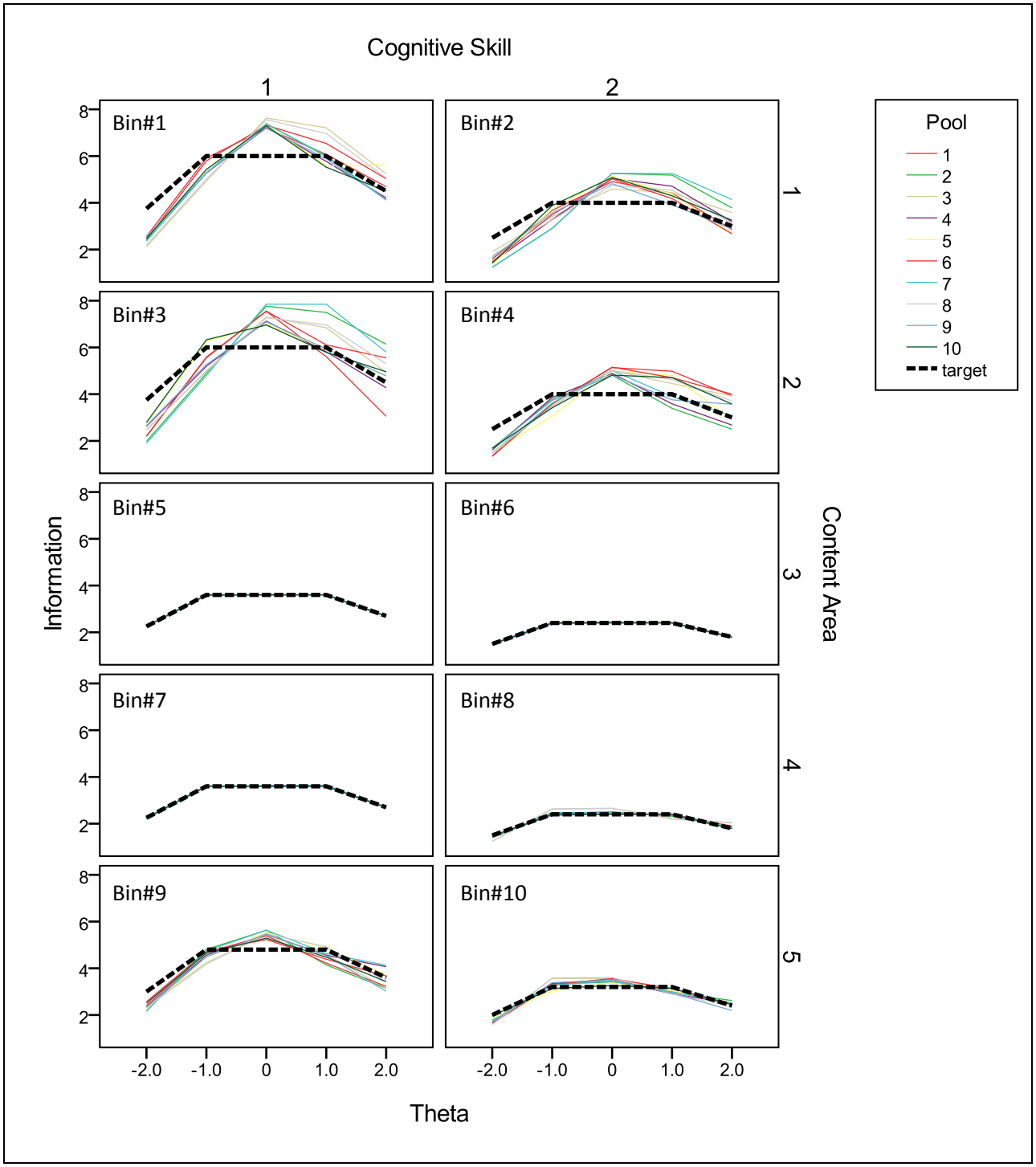


Figure A.2. Pool Information Functions for 10 Pools Constructed With MSDM (MIQP)