Measurement and Research Department Reports 2005-1

An Approximation of Cronbach's α and its Use in Test Assembly

Angela J. Verschoor

cito groep

Measurement and Research Department Reports 2005-1

An Approximation of Cronbach's α and its Use in Test Assembly

Angela J. Verschoor

Cito
Arnhem, 2005

Abstract

In this paper a new approximation of Cronbach's α is presented. It is especially suited in the context of test assembly. Using this approximation, two test assembly models are introduced. Being non-linear models, they are solved by Genetic Algorithms as the commonly used Linear Programming methods cannot be used here. A comparison is made with existing test assembly models.

1 Introduction

Test assembly modeling for Classical Test Theory has been a somewhat neglected field. It concerns the selection of items from a pool in such a way that the reliability of the resulting test is maximised under conditions like a fixed test length, a taxonomic make up, and other demands. Assembling a test with maximum reliability involves non-linear models, difficult to solve using traditional optimisation techniques. Adema and Van der Linden (1989) were the first to circumvent this difficulty by formulating a linearisation of the test's reliability. Armstrong, Jones and Wang (1994) took a different approach by formulating the problem as a network flow model. Both models can be solved using standard Linear Programming methods, but are prone to yielding suboptimal solutions or are applicable in specific contexts only.

In this paper, two test assembly models are introduced that use an approximation of Cronbach's α as an operationalisation of the reliability of the test. Utilising optimisation techniques like Genetic Algorithms, introduced by Holland (1975), it is possible to solve this class of models efficiently without loss of generality as in the network flow models of Armstrong et al. Genetic Algorithms (GAs) are members of a class of local search heuristic that can guarantee to converge to the optimal solution, although a specific solution cannot be proven to be optimal (cf. Eiben, Aarts and Van Hee, 1991). GAs optimise models by emulating principles from evolution theory in biology. They process a population of solutions to the problem iteratively. In each iteration, a mating process creates offspring by the use of operators as *crossover* and *mutation*. Subsequently a 'survival of the fittest' rule is applied in such a way that the fittest solutions survive to the next generation while the population size remains constant. Instrumental in these mating and survival processes is the *fitness function*. The higher a solution's fitness, the higher its probability to procreate and survive. It has a relation with the objective function: both optima must coincide. But, as the crossover and mutation can generate any solution, the fitness function value of the optimum must be higher than the fitness of all infeasible solutions. As the mating process can create a wide variety of solutions, infeasibility is dealt with by means of a penalty function not unlike Lagrangian Relaxation. In concordance with the findings of Siedlecki and Sklanski (1989), a dynamic penalty adaptation is used to establish the optimal values for the penalty multipliers.

Furthermore, in simulation studies several penalty adaptation schemes are investigated and the results are compared to solutions of an existing test assembly model.

2 CMAX Model

In test assembly using Classical Test Theory a key notion is the reliability of the to-be-assembled test or, more specifically, its lower bound Cronbach's coefficient

α:

$$\alpha = \frac{k}{k-1}\left\{1 - \frac{\sum_i \sigma_i^2}{\sigma_X^2}\right\} = \frac{k}{k-1}\left\{1 - \frac{\sum_i \sigma_i^2}{\left(\sum_i \rho_{it}\sigma_i\right)^2}\right\} \tag{1}$$

where k is the test length, σ_i^2 is the variance of item i, and ρ_{it} the correlation between the item score and test score. Using equation (1), a test assembly model can be formulated for situations in which the test assembler wishes to assemble a test with maximum α while adhering to test specifications regarding limited resources like test length or test time, a desired difficulty range, a taxonomic makeup and restrictions at the item level:

$$\text{maximise} \quad \alpha = \frac{\sum_i x_i}{\sum_i x_i - 1}\left\{1 - \frac{\sum_i \sigma_i^2 x_i}{\left(\sum_i \rho_{it}\sigma_i x_i\right)^2}\right\} \tag{2}$$

$$\text{subject to:} \quad P^\ell \leq \frac{\sum_i \pi_i x_i}{\sum_i x_i} \leq P^u \tag{3}$$

$$\sum_i q_{in} x_i \leq Q_n \qquad \forall n \tag{4}$$

$$C_m^\ell \leq \sum_i c_{im} x_i \leq C_m^u \qquad \forall m \tag{5}$$

$$p_r(x) = 1 \qquad \forall r \tag{6}$$

$$x_i = \begin{cases} 1, & \text{item } i \text{ in the test} \\ 0, & \text{else} \end{cases} \qquad \forall i$$

Variables x_i are the decision variables indicating whether an item is selected or not. P^ℓ and P^u restrict the test's difficulty. Coefficients q_{in} are the item's resource parameters. These parameters define how many resources, for example test time, will be needed to include the item in the test. Coefficients c_{im} are the classification parameters having value 1 if item i belongs to category m and 0 otherwise. Q_n, C_m^ℓ and C_m^u are the test's desired use of resources and number of items representing the classification categories, respectively. Equation (6) expresses relations on the item level, the inter item relations. Common examples of inter item relations are enemy sets and testlets. Enemy sets are groups of mutually excluding items while testlets are groups of mutually including items. More general relations can be formulated as well, using Boolean operators $\vee$, $\wedge$ and $\neg$. These can be transformed into numerical expressions using MIN and MAX. Functions $p_r(x)$ are the payoff functions for the inter item relations, according to De Jong and Spears (1989). The payoff functions are derived from relations by using AVE to evaluate $\wedge$ clauses and MAX for $\vee$ clauses. However, the only type of relations that can successfully be treated are the ones in which the $\neg$ -operator has only the scope of individual variables. Conversion from other types of relations is simply a matter of applying De Morgan's laws.

In practical situations a problem arises as equation (2) can only be evaluated after the test has been administered. Furthermore, even if equation (2) could

be evaluated, it would result in a non-linear model that would need special techniques to be solved. Therefore Adema and Van der Linden (1989) proposed their Model II in which equation (2) is replaced by:

$$\text{maximise} \quad \sum_i r_{it} x_i \tag{7}$$

where r_{it} is the point-biserial correlation between item score i and the score of the test in which the item was pretested. Their model is based on Gulliksen's observation (1950) that, in general, in equation (1), the sum of item variances varies less than the test variance, and so α can be expected to depend more on the latter. This effect is also verified by Ebel (1967) and suggests that it is more profitable to select items with a high r_{it} than with a low one, and thus that it is most efficient to use (7) as the objective.

Adema and Van der Linden's model has the advantage that it is linear, and thus can be optimised by the use of Linear Programming techniques. In its original form, however, the model has a drawback. The α of the to-be-assembled test cannot be determined beforehand. During test assembly, the test to be assembled is not known, and neither are the corresponding r_{it}'s. This drawback can be overcome by the assumption that the r_{it}'s of the newly assembled test will have a high correlation with the r_{it}'s of previous pretests.

But in general the item pools used for test assembly consist of items originating from different test forms. If the test forms used in building the item pool vary in length, items from shorter test forms are favoured without proper ground for the following reason given by Guilford (1954, p.439):

> When an item is correlated with the total score of which it is a part, the value of r_{it} tends to be inflated. The shorter the test, the greater this inflation is likely to be. Even if all items correlated actually zero with what the total score measures, and if all item variances were equal, each item would correlate to the extent of $1/\sqrt{n}$, where n is the number of items.

In order to overcome this bias, item discrimination indices are needed that are independent from test contexts such as test length and test variance. Zubin (1934) faced this problem by concentrating on the item-rest correlation r_{ir}. This would mean, however, that the items would be correlated with slightly different remainders. It is also easy to see that if items are positively correlated with the remaining items, coefficient r_{ir} is not invariant to test length either. This would introduce a new spuriousness, now favouring items originating from longer tests.

Consider the point-biserial correlations found in earlier analyses, corrected for unique item variances by Henrysson (1963):

$$h_{it} = \sqrt{\frac{k}{k-1}} \frac{r_{it} s_X - s_i}{\sqrt{s_X^2 - \sum_i s_i^2}} \tag{8}$$

In equation (8), coefficient h_{it} is claimed to be invariant to test length. In order to verify this claim, a simulation was conducted. An item bank consisting

of 500 items was used with simulated IRT-parameters according to the two-parameter model with $\log(\alpha) \sim N(0, 0.4)$ and $\beta \sim N(0, 1)$. A population of 100,000 simulated candidates with $\vartheta \sim N(0.07, 0.35)$ was used to generate item scores. From these data, several (sub)tests with varying length were defined and analysed. In Table (1), the definition of the tests is given, as well as the r_{it}, r_{ir} and h_{it} for item 1.

Table 1: Analysis of r_{it}, r_{ir} and h_{it} for item 1

	items	r_{1t}	r_{1r}	h_{1t}
test 1	1..5	0.477	0.040	0.152
test 2	1..10	0.311	0.050	0.145
test 3	1..20	0.256	0.074	0.149
test 4	1..40	0.214	0.102	0.151
test 5	1..80	0.184	0.119	0.149
test 6	1..500	0.155	0.143	0.149

It is clear that even for tests of length 80 the bias of r_{it} and r_{ir} is still large, while even for short tests the h_{it} remains relatively stable. Therefore h_{it} can be regarded as an item discrimination index free from the context of the test in which it was originally included. Thus, if the item was included in several tests administered in the same population, all h_{it}'s are estimates of the same entity h_i, the item's true discrimination. Its square, h_i^2, is an estimate of the item communality in a factor analysis. Using h_i instead of r_{it} removes spurious preference towards items analysed in short tests. Substituting h_i for r_{it} in (7) overcomes the bias caused by the context of the pretests, but prediction of α of the newly assembled test remains cumbersome. Calculation of new point-biserials using equation (8) is not possible since the test variance is not known.

Now, consider the test variance to be formulated as:

$$\sigma_X^2 = \sum_i \sigma_i^2 + \sum_{i \neq j} \mathrm{Cov}(X_i, X_j)$$

From the assumption that h_i is regarded as the factor loading in unidimensional data, it can be observed that $\forall i, j : \mathrm{Cov}(X_i, X_j) = h_i \sigma_i h_j \sigma_j$. From this observation, the following relation can be derived as an estimate for Cronbach's α for a newly assembled test:

$$\alpha^* = \frac{\sum_i x_i}{\sum_i x_i - 1} \left\{ 1 - \frac{\sum_i \sigma_i^2 x_i}{\sum_i \sigma_i^2 x_i + \sum_{i \neq j} h_i \sigma_i h_j \sigma_j x_i x_j} \right\} \tag{9}$$

Using equation (9), the CMAX model is formulated as:

$$\text{maximise} \quad \frac{\sum_i x_i}{\sum_i x_i - 1}\left\{1 - \frac{\sum_i \sigma_i^2 x_i}{\sum_i \sigma_i^2 x_i + \sum_{i \neq j} h_i \sigma_i h_j \sigma_j x_i x_j}\right\} \tag{10}$$

$$\text{subject to:} \quad P^\ell \leq \frac{\sum_i \pi_i x_i}{\sum_i x_i} \leq P^u$$

$$\sum_i q_{in} x_i \leq Q_n \qquad \forall n$$

$$C_m^\ell \leq \sum_i c_{im} x_i \leq C_m^u \qquad \forall m$$

$$p_r(x) = 1 \qquad \forall r$$

$$x_i = \begin{cases} 1, & \text{item } i \text{ in the test} \\ 0, & \text{else} \end{cases} \qquad \forall i$$

As the objective in the CMAX model (10) is non-linear, traditional methods based upon Linear Programming are unsuitable. On the other hand, Verschoor (2004) has shown that Genetic Algorithms (GA) are capable of solving this class of optimisation problems efficiently.

The first step in solving optimisation problems using GAs is to formulate a suitable fitness function. Similar to the IMAX model of Verschoor (2004) the fitness function for the CMAX model $f(x)$ is defined by α^*, compensated with a penalty function $g(x)$ dependent on the rate in which the restrictions are exceeded:

$$f(x) = \frac{\alpha^*}{1 + g(x)} \tag{11}$$

$$\begin{aligned} g(x) = &\kappa\, h\left(P^\ell - \frac{\sum_i p_i x_i}{\sum_i x_i}\right) + \kappa\, h\left(\frac{\sum_i p_i x_i}{\sum_i x_i} - P^u\right) \\ &+ \lambda \sum_n h\left(\sum_i q_{in} x_i - Q_n\right) \\ &+ \mu \sum_m h\left(C_m^\ell - \sum_i c_{im} x_i\right) + \mu \sum_m h\left(\sum_i c_{im} x_i - C_m^u\right) \\ &+ \varphi \sum_r 1 - p_r(x) \end{aligned}$$

$$h(u) = \begin{cases} u, & u > 0 \\ 0, & u \leq 0 \end{cases}$$

Coefficients κ, λ, μ and φ are the penalty multipliers that are determined dynamically. Consider for k consecutive iterations the individuals with the highest fitness. If for all these k individuals all resource restrictions are met, multiply λ by $1-\delta$. If for all individuals some resource restrictions are not met, multiply λ by $1+\varepsilon$. Leave λ unchanged in other cases, that is, that for some individuals all resource restrictions are met while for other individuals there are offending restrictions. The same rule is followed for the other penalty multipliers κ, μ and φ. Usually, k has a fixed value in the order of magnitude of 10 - 40 iterations.

2.1 Simulations

Simulations were conducted in two phases. In the first phase, the dynamic penalty scheme was investigated while in the second phase several stopping criteria were considered.

The question to be answered in the first phase is what values should k, δ and ε assume in order to give a good performance. Specifically, can a fixed set of values be found or is there a dependency with model parameters such as its complexity expressed in the number of restrictions?

To answer this question three different test assembly models were used. They were based upon the same item bank used in the simulations of the h_i-coefficients. Furthermore, all items were classified on three different arbitrary classifications. The first classification consisted of 4 categories 101, 102, 103, and 104. All items were uniformly distributed over these categories. The second classification consisted of 4 categories 201, 202, 203, and 204. These categories were furnished with approximately 250, 125, 85 and 40 items, respectively. The third classification contained categories 301..310, to which the items were assigned uniformly.

The simplest test assembly model, 1, had one restriction related to the target test difficulty: $P^{\ell} = 0.6 \leq \frac{\sum_i \pi_i x_i}{\sum_i x_i}$, and one resource restriction: $\sum_i x_i \leq 40$. No content restrictions based on the classification structure described above were used. Model 2 was an extension of 1, having a test grid with the columns representing categories 101..104, the rows representing categories 201..204, and each cell requiring a minimum of 2 items and a maximum of 3 items. Model 3 had a more extensive test grid consisting of the categories 201..204 in the columns and categories 301..310 in the rows. Each cell in the test grid was required to contain exactly one item.

Six variable penalty schemes were investigated. Three iteration cycles with $k = 10$ (fast), $k = 20$ (moderate), and $k = 40$ (slow) were combined with two adaptation schemes with $\delta = 0.11, \varepsilon = 0.08$ (high), and $\delta = 0.03, \varepsilon = 0.02$ (low). The algorithm was stopped after 4000 iterations for model 1, 8000 iterations for 2, and 32000 iterations for model 3. In Table (2), the best solutions are reported as well as the iterations in which they were found.

In Table (2) it can be seen, for example, that for the fast and low adaptation scheme applied on model 3, an average α^* of 0.8017 was found in on average

Table 2: Variable penalty schemes

		low adaptation		high adaptation	
		α^*	iteration	α^*	iteration
1	fast	0.8492	400 ± 130	0.8492	250 ± 90
	mod.	0.8492	450 ± 220	0.8492	330 ± 90
	slow	0.8492	420 ± 190	0.8492	410 ± 140
2	fast	0.8316 ± 0.0007	3500 ± 2200	0.8316 ± 0.0007	3300 ± 2300
	mod.	0.8317 ± 0.0006	3500 ± 2100	0.8316 ± 0.0006	3100 ± 2300
	slow	0.8316 ± 0.0007	3800 ± 2000	0.8317 ± 0.0006	3200 ± 2200
3	fast	0.8017 ± 0.0028	21700 ± 7700	0.8009 ± 0.0035	22700 ± 6900
	mod.	0.8021 ± 0.0027	21400 ± 7600	0.8011 ± 0.0035	21300 ± 8100
	slow	0.8025 ± 0.0024	21100 ± 7600	0.8018 ± 0.0028	21900 ± 7600

21700 iterations. The conclusion can be drawn that the more complex a model is, the better a low and slow adaptation scheme seems to perform. As the number of iterations needed tends to grow with the model complexity, the low and slow scheme could be selected in all cases without great loss of efficiency for the less complex models.

In the second phase the question was if a stopping rule could be devised. Two stopping rules, both based on the number of items and the number of restrictions in the models, were investigated. A fast rule stopped after IK iterations, where I is the number of items in the pool and K the total number of restrictions in the model instance. The second rule, a slower one, generally giving better solutions, stopped at $IK \ln(IK)$ iterations. A slow and low penalty adaptation scheme was used for these simulations.

In Table (3), the α^* reached at the stopping criteria are reported as well as the optimal values for α^*, and the deviations in terms of extra items needed to extend the test in order to reach the optimal α^* according to the Spearman-Brown formula for test lengthening.

Table 3: Optimal reliabilities and percentages of deviation

Model	stop at iteration		α^* reported	optimal α^*	deviation
1	IK	1000	0.8492	0.8492	–
	$IK \ln(IK)$	6908	0.8492		–
2	IK	17000	0.8318 ± 0.0005	0.8321	0.2%
	$IK \ln(IK)$	165596	0.8321 ± 0.0002		$< 0.1\%$
3	IK	41000	0.8030 ± 0.0022	0.8057	1.7%
	$IK \ln(IK)$	435474	0.8054 ± 0.0005		0.2%

When adhering to the norm of stopping when a solution is found within a 2%-bandwidth from the optimum, the first stopping rule will generally be sufficient. As no indication can be given that such a solution has been found, this situation

cannot be assumed for all individual cases, but only for the averages.

3 Comparison of CMAX and Model II

With the simulated item bank described above, the CMAX model and Model II of Adema and Van der Linden were compared in order to investigate whether the models give different solutions. For this purpose, models 1 and 3 were adapted for both test assembly models. Model 1 for CMAX was compared to the adapted model 1 for Model II. Both models designated a test as the optimal one, and for these two tests new response data for the simulated population was generated and analysed. The model 1 for CMAX and model 1 for Model II have

Table 4: Comparison of Model II and the CMAX model

Model	$\sum r_{it}$	α^*	α	deviation $\alpha - \alpha^*$
1-Model II	13.95	0.8492	0.8509	1.3%
1-CMAX	13.95	0.8492	0.8509	1.3%
deviation		–	–	
3-Model II	12.03	0.8034	0.8051	1.1%
3-CMAX	12.00	0.8057	0.8072	1.0%
deviation		1.5%	1.3%	

the same optimum. After analysis of the new response data for this optimal test, Cronbach's α was observed to be 0.8509, while α^* was 0.8492, a slight underestimation of 1.3%, in the order of magnitude of half an item difference in test length. For model 3, the same procedure was followed. CMAX and Model II have different optimal solutions in the case of model 3. The CMAX optimum has an α^* is approximately 1.5% better than the Model II optimum according to the Spearman and Brown formula.

For these solutions too new response data were generated and analysed. As with model 1, α^* was smaller than the observed α. In first instance this might suggest that α^* is a lower bound for Cronbach's α, and therefore for the test reliability, but this cannot be guaranteed in general. But nonetheless, differences are relatively small, so that a new test assembly model becomes feasible.

4 CMIN model

The CMIN model (12) expresses the wish to construct a test that has a minimal use of resources, given a threshold reliability α_T, a desired difficulty, a set of resource restrictions, a desired content balancing and inter item restrictions. It can be applied in situations that require a test with a reliability at least as high as the target α_T, to be realised against minimal cost, expressed by, for example, test length. It is formulated as:

$$
\begin{aligned}
\text{minimise} \quad & \sum_i q_{i0} x_i && (12)\\
\text{subject to:} \quad & \alpha^* \geq \alpha_T \\
& P^\ell \leq \frac{\sum_i \pi_i x_i}{\sum_i x_i} \leq P^u \\
& \sum_i q_{in} x_i \geq Q_n && \forall n \\
& C_m^\ell \leq \sum_i c_{im} x_i \leq C_m^u && \forall m \\
& p_r(x) = 1 && \forall r \\
& x_i \in \{0,1\} && \forall i
\end{aligned}
$$

Note that as α^* is not a lower bound for α, the observed α need not be larger than α_T. As with the fitness functions of the Genetic Algorithms solving the CMAX model and the IMAX model, the fitness function for the CMIN model bears a great resemblance to the fitness function for the IMIN model. Here again, the penalty function is extended with a term to accommodate the difficulty restrictions. Similar to the situation in the IMIN model, a reward is added for surplus reliability:

$$
f(x) = \left(\frac{1}{1+\sum_i q_{i0} x_i}\right)\left(\frac{1}{1+g(x)}\right) \tag{13}
$$

$$
\begin{aligned}
g(x) =& \kappa\ h\left(\alpha_T - \alpha^*\right) + \kappa\ h\left(P^\ell - \frac{\sum_i \pi_i x_i}{\sum_i x_i}\right) + \kappa\ h\left(\frac{\sum_i \pi_i x_i}{\sum_i x_i} - P^u\right) \\
& + \lambda \sum_n h\left(Q_n - \sum_i q_{in} x_i\right) \\
& + \mu \sum_m h\left(C_m^\ell - \sum_i c_{im} x_i\right) + \mu \sum_m h\left(\sum_i c_{im} x_i - C_m^u\right) \\
& + \varphi \sum_r 1 - p_r(x)
\end{aligned}
$$

Especially when using the test length as the objective, there are many solutions with equal objective function values. Since all feasible solutions with the same objective function value will have the same fitness, *epistasis* is the result. Epistasis is the situation in which a GA cannot make a distinction between solutions and the process will stagnate because it lacks a search direction. An obvious way to reduce the epistasis is to allow small differences in fitness so that every solution has a unique fitness. These differences should not be assigned randomly

to the solutions. They should have some meaningful value in order to accelerate the optimisation process.

It is easy to see that, given two feasible solutions, it is easier to remove an item from the most reliable test than from the least reliable test while retaining feasibility. Thus a reward γ should be assigned to each solution for each surplus unit of α^*, and therefore the fitness function in equation (13) should be replaced by equation (14):

$$f(x) = \left(\frac{1}{1+\sum_i q_{i0}x_i}\right)\left(\frac{1+\gamma h(\alpha^* - \alpha_T)}{1+g(x)}\right) \tag{14}$$

The reward, however, should not be too high. If adding an item to a feasible solution is rewarded in such a way that it compensates for the increase in objective function and subsequent loss in fitness, the optimum of the fitness function does not coincide with the optimum of the objective function anymore. If the removal of an item results in a feasible solution, it must always be more profitable than the reward for the surplus of reliability caused by keeping the item in the test.

4.1 Simulations

In order to investigate the CMIN model, simulations were performed. Three models were simulated using the same item bank as for the CMAX models: models 4, 5 and 6 respectively. Model 4 had one restriction pertaining to the threshold reliability: $\alpha_T = 0.83$, and one restriction pertaining to the target test difficulty: $P^\ell = 0.6 \leq \frac{\sum_i \pi_i x_i}{\sum_i x_i}$. For model 5 two sets of classification restrictions were defined: for categories 201..204 the maximum percentage of representation was 55, 30, 20, and 10, while for categories 301..310 the maximum percentages were 20. Model 6 had the same test grid as model 3 in the simulations of the CMAX model, consisting of the categories 201..204 in the columns and categories 301..310 in the rows. Each cell in the test grid was required to contain a maximum of one item. At the same time, the threshold α_T was lowered to 0.80 as a test of $\alpha^* \geq 0.83$ appeared not to be feasible in combination with the test grid. Note that for model 5 the classification restrictions in the CMIN model needed a small modification in order to accommodate a representation in terms of percentages of test length instead of a fixed number of items:

$$C_m^\ell \leq \frac{\sum_i c_{im}x_i}{\sum_i x_i} \leq C_m^u \qquad \forall m \tag{15}$$

Two fixed stopping rules were applied, a fast rule stopping at IK iterations and a slower one stopping at $IK\ln(IK)$ iterations. Furthermore, a slow and low adaptation scheme was used with $k = 40$, $\delta = 0.03$ and $\varepsilon = 0.02$.

Only for model 4 and the fast stopping rule, the optimal test length of 28 items was occasionally not reached and a test of length 29 was assembled. In all other cases the optimal test lengths were reached.

Table 5: Optimal solutions for the CMIN models

Model	stop at iteration		α^*	fitness*100	test length
4	IK	1000	0.8306	3.4404	28.07
	$IK\ln(IK)$	6908	0.8303	3.4484	28
5	IK	17000	0.8309	3.3336	29
	$IK\ln(IK)$	165596	0.8309	3.3336	29
6	IK	41000	0.8020	4.0008	24
	$IK\ln(IK)$	435474	0.8020	4.0008	24

5 Conclusions

While Cronbach's α is an important and easy to understand concept, it cannot be evaluated during test assembly. Current test assembly models do not attempt to evaluate α, although they try to optimize it. This circumvention leads necessarily to suboptimal solutions.

This paper introduces an approximation of α, α^*, that can be evaluated during test assembly. The price one has to pay is the assumption of a unidimensional item pool, an often implicitly made assumption. As the resulting test assembly model, CMAX, is non-linear, Genetic Algorithms are used. Comparison of a test assembly model using the approximation α^* with Adema and Van der Linden's Model II shows the benefit of this approach. A second advantage of α^* is that a new test assembly model becomes available: the CMIN model minimizes the resources needed for the test given a minimum α^*. The CMIN model finds its justification in those situations in which circumstances dictate the realisation of a test with a specific reliability, or higher. The test with minimal cost that has such a reliability will be found.

References

[Adema and van der Linden, 1989] Adema, J. and van der Linden, W. (1989). Algorithms for computerized test construction using classical item parameters. *Journal of Educational Statistics*, 14:279–290.

[Armstrong et al., 1994] Armstrong, R., Jones, D., and Wang, Z. (1994). Automated parallel test construction using classical test theory. *Journal of Educational Statistics*, 19:73–90.

[De Jong and Spears, 1989] De Jong, K. and Spears, W. (1989). Using genetic algorithms to solve NP-complete problems. In Schaffer, J., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 124–132. San Mateo, CA: Morgan Kaufmann.

[Ebel, 1967] Ebel, R. (1967). The relation of item discrimination to test reliability. *Journal of Educational Measurement*, 4:125–128.

[Eiben et al., 1991] Eiben, A., Aarts, E., and van Hee, K. (1991). Global convergence of genetic algorithms: a Markov chain analysis. In Schwefel, H.-P. and Männer, R., editors, *Parallel Problem Solving from Nature I*, pages 4–12. Berlin: Springer.

[Guilford, 1954] Guilford, J. (1954). *Psychometric Methods.* New York: McGraw-Hill, 2nd edition.

[Gulliksen, 1950] Gulliksen, H. (1950). *Theory of Mental Tests.* New York: Wiley.

[Henrysson, 1963] Henrysson, S. (1963). Correction of item-total correlations in item analysis. *Psychometrika*, 28:211–218.

[Holland, 1975] Holland, J. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press.

[Siedlecki and Sklanski, 1989] Siedlecki, W. and Sklanski, J. (1989). Constrained genetic optimization via dynamic reward-penalty balancing and its use in pattern recognition. In Schaffer, J., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 141–150. San Mateo, CA: Morgan Kaufmann.

[Verschoor, 2004] Verschoor, A. (2004). IRT test assembly using genetic algorithms. Technical Report Measurement and Research Department Reports 2004-4, Arnhem: Cito.

[Zubin, 1934] Zubin, J. (1934). The method of internal consistency for selecting test items. *Journal of educational Psychology*, 25:345–356.