The Rasch Model and various extensions of this model can be formulated as a quasi loglinear model for the incomplete subgroup x score x item response 1 x item response k contingency table. By comparing various loglinear models, specific deviations of the Rasch model can be tested. Parameter estimates can be computed using programs such as GLIM, ECTA, and MULTIGUAL, but this becomes impractical if the number of items is large. In that case, the tables of observed and expected counts become too large for computer storage. In this paper, a method of parameter estimation is described that does not require the internal representation of all observed and expected counts, but rather uses only the observed and expected sufficient statistics of the parameter estimates, which are the marginal tables corresponding to the model terms only. The computational problem boils down to computation of the expected sufficient statistics which, in its raw form, amounts to summation of a very large number of expected counts. However, it is shown that, depending on the structure of the model, the number of computations can be reduced considerably by making use of the distributive law. As a result, simpler models may be computed much more efficiently in terms of both storage and processing times. Three data tables are provided. (Author/TJH)
Estimating Quasi-Loglinear Models for a Rasch Table if the Numbers of Items is Large

Henk Kelderman

University of Twente
Estimating Quasi-Loglinear Models for the Rasch Table if the Number of Items is Large

Henk Kelderman
Abstract

The Rasch model and various extentions of this model can be formulated as a quasi loglinear model for the incomplete subgroup x score x item response 1 x ... x item response k contingency table. By comparing various loglinear models, specific deviations of the Rasch model can be tested. Parameter estimates can be computed using programmes such as GLIM, ECTA and MULTIQUAL, but this becomes unpractical if the number of items is large. In that case the tables of observed and expected counts become too large for computer storage.

In this paper a method of parameter estimation is described that does not require the internal representation of all observed and expected counts but uses only the observed and expected sufficient statistics of the parameter estimates which are the marginal tables corresponding to the model terms only. The computational problem boils down to computation of the expected sufficient statistics which in its raw form amounts to summation of a very large number of expected counts. It is shown, however, that depending on the structure of the model, the number of computations can be reduced considerably by making use of the distributive law. So that simpler models may be computed much more efficiently both in terms of storage and processing time.

Keywords: Rasch model, Quasi-Loglinear Model, Incomplete Contingency Table, Sufficient Statistics.
Estimating Quasi-Loglinear Models for the Rasch Table if the Number of Items is Large

Contingency table methods (Andersen, 1980; Bishop, Fienberg, & Holland, 1975; Fienberg, 1980; Gokhale, & Kullback, 1978; Goodman, 1978; Haberman, 1978, 1979) have been used to estimate and test various psychometric models. Loglinear models -- or their multiplicative equivalent, with or without unobserved variables (Haberman, 1979) -- have been applied to Guttman's (1950) perfect scale model (Clogg & Sawyer, 1981; Divison, 1980; Dayton & Macready, 1980; Goodman, 1959, 1975), to models for mastery tests (Macready & Dayton, 1977; Bergan, Cancelli and Luiten, 1980; van der Linden, 1980), to a model of item homogeneity (Lienert, & Raatz, 1981) to Coombs' (1964) unfolding model (Davison, 1979), and to the Bradley-Terry model for paired comparisons (Fienberg, 1980).

Mellenbergh, and Vijn (1981) noticed the close similarity between the Rasch (1960, 1966) model and a loglinear model for the score x item number x item response contingency table. Vijn and Mellenbergh (1982) showed that the parameter estimates of this model are identical to Wright and Panchapakesan's (1969) unconditional maximum likelihood (UML) estimates of the parameters in the Rasch model.

Tjur (1982) showed that the conditional Rasch model can be formulated as a multiplicative Poisson model for an item 1 x...x item k contingency table, where k is the number of items. For the same contingency table, Cressie and Holland (1983) formulated a loglinear
Rasch model without conditioning on the sum score. The model is not a standard loglinear model since it includes interaction terms at every possible level. Moreover, because there is no conditioning on the sum score, the distribution of the latent trait must be taken into account. To ensure that this distribution exists, complicated inequality constraints must be imposed on the interaction terms.

Kelderman (1984) formulated the conditional Rasch model as a standard quasi-loglinear model for the incomplete score \( x \) item 1 \( x \) ... \( x \) item \( k \) contingency table. In the sequel of this paper, this table is called the "Rasch table". By adding a sum score way to the table, the interaction terms vanish and the model becomes a quasi-independence model with main effects for the item responses only. Using ordinary incomplete-contingency-table methods the conditional Rasch model can be estimated and the overall goodness of fit tested. Moreover, the model can be tested against less restrictive quasi-loglinear models to detect specific deviations from Rasch homogeneity.

The general quasi-loglinear model for the Rasch table is as follows. Let \( x \) denote the vector of variables \( (x_1, \ldots, x_{k+1}) \), where the first \( k \) variables are the responses to dichotomous items, scored zero for an incorrect response and one for a correct response, and the \( k + 1 \)th variable is the sum score. Let all the \( k + 1 \) possible indices of the variables be collected in the set \( U = \{1, \ldots, k+1\} \).

Furthermore, let \( A_1, \ldots, A_\lambda, \ldots, A_s \) be subsets of \( U \). Each set \( A_\lambda \) corresponds to a model term in the quasi-loglinear model described below. The elements of \( A_\lambda \) are the numbers of the variables on which these model terms depend. The vector of these variables is denoted as \( x_{A_\lambda} \), i.e. it is the vector of the variables \( x_j \) where the variable
numbers $j$ are in the set $A_x$. For example if $A_x = \{1,3,4\}$ then $x_A = (x_1, x_3, x_4)$. Using this notation, a quasi-loglinear model for the Rasch table can be written as:

\[ \ln m_x = u_{A_1}(x_{A_1}) + \ldots + u_{A_s}(x_{A_s}), \]

with constraints

\[ \sum_{j=0}^{1} u_{A_j}(x_{A_j}) = 0, \]

for all $j \in A_x$ with $x = 1, \ldots, s$, where $x_1 = 0,1; \ldots; x_k = 0,1; x_{k+1} = x_1 + \ldots + x_k$ and where $\ln$ is the natural logarithm and $m_x$ is the expected number of individuals in cell $x$. The $u$-terms $u_{A_1}(x_{A_1})$, ..., $u_{A_s}(x_{A_s})$ denote grand mean, main effects and interaction effects of the variables $x_1, \ldots, x_{k+1}$ like in ordinary ANOVA models. The models must conform to the hierarchy principle, that is, if a $u$-term occurs in the model all lower order relatives must also occur in the model.

In this paper the equivalent multiplicative form of the general loglinear model (1) is used:

\[ m_x = \phi_{A_1}(x_{A_1}) \ldots \phi_{A_s}(x_{A_s}) \]

where

\[ \phi_{A_j}(x_{A_j}) = \exp(u_{A_j}(x_{A_j})). \]
Kelderman (1984) showed that, if the sumscore variable is considered a fixed variable, and the item responses as random variables, the quasi-independence model (Goodman, 1968), that is, a model with main effects only, is equivalent to the conditional Rasch model. The model has \( A_1 = 0, A_2 = \{1\}, \ldots, A_{k+2} = \{k+1\} \) so that:

\[
\ln m_{x_1 \cdots x_{k+1}} = u + u_2(x_2) + \cdots + u_k(x_k) + u_{k+1}(x_{k+1}),
\]

with the constraints above. The first item parameter is set to zero to fix the scale (Kelderman, 1984).

The advantage of using loglinear models for Rasch analysis is that there is a great flexibility in the testing of Rasch models against various alternative quasi loglinear models, and estimating the parameters of these models.

To date, the disadvantage of loglinear models is, however, that only a limited number of items and responses can be analyzed simultaneously. The models we work with are defined for an incomplete score \( x \) item 1 \( x \) \( \ldots \) item \( k \) contingency table. The number of cells of this table is the product of the numbers of categories of each of the ways. If the number of items or subgroups becomes large, the table becomes very large and, for the usual data sets, many of the structurally non-zero cells of the observed table will become empty.

Most present-day computer programs for the analysis of contingency tables by loglinear models (Baker & Nelder, 1978; Goodman & Fay, 1974) require the internal storage of the observed and expected table of counts, which is virtually impossible for the present problem. However, the table need only exist in theory. Firstly, the data can
be stored in an ordinary subjects x variables data matrix, which avoids storage of structural and nonstructural zero's. Secondly, the parameter estimates may be calculated by solving the likelihood equations in terms of the set of minimal sufficient statistics. For the Rasch model, very efficient algorithms to solve the likelihood equations in terms of minimal sufficient marginals have been developed by Andersen (1972). Andersen's algorithms can also be applied to models that can be broken down into a set of separate Rasch models, e.g., models with different item parameters, within each subgroup, within each scoregroup, or both (see also Andersen, 1980a, p. 251). For other models, e.g., models with parameters describing interactions between item responses the algorithm is of limited use.

Goodman (1964, 1968) describes an algorithm for the analysis of incomplete tables by quasi-loglinear models that calculates the parameter estimates from sufficient statistics. He uses the likelihood equations

\[ f_{x_{A_i}} = m_{x_{A_i}} \]

for all \( x_{A_i}, i = 1, \ldots, s \), where \( f_{x_{A_i}} \) denotes the observed marginal counts for the value \( x_{A_i} \) of variables \( A_i \) and \( m_{x_{A_i}} \) the corresponding expected marginal counts.

The parameter \( \Phi_{A_i}(x_{A_i}) \) can be derived from (4) by writing the expected marginal counts in terms of the parameters

\[ f_{x_{A_i}} = \Phi_{A_1}(x_{A_1}) \cdots \Phi_{A_s}(x_{A_s}) \]
where $\overline{A}_i$ is the set of variables not in $A_i$ (i.e. $U-A_i$). That is, the expected marginal counts $m_{\overline{A}_i}$ are obtained by summing the expected marginal cell counts which depend on $x = (x_{A_i}, x_{\overline{A}_i})$ over all values of $x_{\overline{A}_i}$. Since $\Phi_{A_i}(x_{A_i})$ does not depend on $x_{\overline{A}_i}$ it can be brought before summation sign and solved as

$$
\Phi_{A_i}(x_{A_i}) = f_{x_{A_i}} \sum_{x_{\overline{A}_i}} \Phi_{A_i}(x_{A_i}) \ldots \Phi_{A_i-1}(x_{A_i-1})
$$

which gives the recursion formula:

$$
\Phi_{A_i}(x_{A_i}) = f_{x_{A_i}} / \sum_{x_{\overline{A}_i}} \Phi_{A_i}(x_{A_i}) \ldots \Phi_{A_i-1}(x_{A_i-1})
$$

which gives the recursion formula:

$$
\Phi_{A_i}(x_{A_i}) = f_{x_{A_i}} / \sum_{x_{\overline{A}_i}} \Phi_{A_i}(x_{A_i}) \ldots \Phi_{A_i-1}(x_{A_i-1})
$$

which gives the recursion formula:

$$
\Phi_{A_i}(x_{A_i}) = f_{x_{A_i}} / \sum_{x_{\overline{A}_i}} \Phi_{A_i}(x_{A_i}) \ldots \Phi_{A_i-1}(x_{A_i-1})
$$

where $r$ denotes the iteration number. This can be used until convergence is reached.

It is easily shown that (5) is equivalent to the iterative proportional fitting algorithm (Bishop, Fienberg, & Holland, 1975, p. 189). Haberman (1974, see Bishop et al, p. 186) gives the conditions under which the iterative proportional fitting algorithm converges to the unique maximum-likelihood estimates for the expected counts.

Estimation of the parameters of quasi-loglinear models for the Rasch table using (5), does not result in storage problems if the
number of items is large. The number of operations, necessary to calculate the expected marginal sums \( m_{A_i} \), however, may still be large. We will now reduce this number of operations so that loglinear models can be more readily applied to Rasch item analysis.

Efficient Computation of Expected Marginal Sums

The marginal table for a set of variables \( A \) with values \( x_A \) is:

\[
(6) \quad m_{x_A} = \sum_{x_A} m_{x_A} = \sum_{x_A} \phi_{A_1}^{A_1}(x_{A_1}) \cdots \phi_{A_s}^{A_s}(x_{A_s})
\]

with the restriction \( x_{k+1} = x_1 + \cdots + x_k \).

Using (6) to compute the marginal counts requires a large number of operations if the number of items is not small: for all possible values of \( x_A \) the product of \( s \) model parameters must be calculated and the results summed. The number of calculations can be reduced, however, by using the distribution law of multiplication over summation and by avoiding repetition of the same calculations.

Before describing this method to calculate the expected sufficient statistics for an arbitrary loglinear model, its principles are illustrated for a small example. It is shown that the expected sum score marginal table of an item 1 x item 2 x item 3 x sum score table can be calculated by repeated multiplication and summation of parameters depending on one item at a time, instead of summing the expected counts over all cells. First it is shown that the number of multiplications can be reduced by multiplying the parameters depending on one item at a time, instead of summing the expected counts over
all cells. First it is shown that the number of multiplications can be reduced by multiplying the parameters depending on the second item with the parameters depending on the first item and using the result in later calculations. Then it is shown that a second summation over the first two items can be accomplished by summing only over the result of the previous multiplications. In addition, the third item can be processed by multiplying the result of the previous summation with the model parameters depending on it and summing the result over item three. Finally the expected sufficient marginal sums are then obtained by multiplying the result with the remaining model parameters depending on the sumscore. The restriction \( x_4 = x_1 + x_2 + x_3 \) is respected, by summing over item 1 (1 - 1, 2, 3) only terms that depend on the same sum score \( t_i = x_1 + \ldots + x_i \) of the first \( i \) items.

As an example consider the case of three items and a model with \( \ell \) main effect parameters and one interaction effect parameter of the third item with the sumscore, i.e., \( k = 3 \) and \( A_1 = \{1\}, A_2 = \{2\}, A_3 = \{3\}, A_4 = \{4\}, A_5 = \{3,4\} \). The multiplicative form of the loglinear model then becomes

\[
(7) \quad m_{x_1x_2x_3x_4} = \phi_1(x_1) \phi_2(x_2) \phi_3(x_3) \phi_4(x_4) \phi_{34}(x_3x_4)
\]

with restriction \( x_4 = x_1 + x_2 + x_3 \). The grand mean effect is left out for simplicity. The expected marginal sum for the sumscore variable \( x_4 \) is:
Estimating Quasi-Loglinear

\[ m_{+++x_4} = \frac{1}{x_1} \cdot \frac{1}{x_2} \cdot \frac{1}{x_3} \cdot x_1^2 \cdot x_2^3 \cdot x_4 \]

\[ x_4 = x_1 \cdot x_2 + x_3 \]

\[ = \frac{1}{x_1} \cdot \frac{1}{x_2} \cdot \frac{1}{x_3} \cdot \phi_1(x_1) \cdot \phi_2(x_2) \cdot \phi_3(x_3) \cdot \phi_4(x_4) \cdot \phi_{34}(x_{34}) \]

\[ x_4 = x_1 \cdot x_2 + x_3 \]

for \( x_4 = 0, 1, 2, 3 \). Writing each of these summations in full:

\[ m_{+++0} = m_{0000} \]

\[ m_{+++1} = m_{1001} + m_{0101} + m_{0011} \]

\[ m_{+++2} = m_{1102} + m_{1012} + m_{0112} \]

\[ m_{+++3} = m_{1113} \]

Applying (8), \( 8 \times 4 = 32 \) multiplications and four additions are required to compute the marginal sums. Considering (8) it is seen that the four products of parameters \( \phi_1(x_1) \) and \( \phi_2(x_2) \) (\( x_1 = 0,1; x_2 = 0,1 \)) are each calculated twice. The number of operations can therefore be reduced by first calculating the product of the parameters depending on variables one or two:

\[ V_2(x_1, x_2) = \phi_1(x_1) \cdot \phi_2(x_2) \]

\( x_1 = 0,1; x_2 = 0,1 \); where the function \( V_2 \) is defined as the product of all model parameters that depend on the first variable (\( x_1 \)), the
second variable \(x_2\), or both. The function \(V_2\) depends on all the variables on which one or more of these parameters depend. In this case there are two main effect parameters so that \(V_2\) depends only on \(x_1\) and \(x_2\). If, however, the model also contained an interaction parameter for the combination of variables \(x_2\) and \(x_3\), \(V_2\) would depend on variables \(x_1\), \(x_2\) and \(x_3\). Later, the \(V_i\) will be defined more generally, but the present definition is consistent with that definition. Substituting (9) in (8) yields:

\[
\begin{align*}
\begin{array}{l}
\quad m_{+++0} = V_2(0,0) \phi_3(0) \phi_4(0) \phi_{34}(00) \\
\quad m_{+++1} = V_2(1,0) \phi_3(0) \phi_4(1) \phi_{34}(01) \\
\quad + V_2(0,1) \phi_3(0) \phi_4(1) \phi_{34}(01) \\
\quad + V_2(0,0) \phi_3(1) \phi_4(1) \phi_{34}(11) \\
\quad m_{+++2} = V_2(1,1) \phi_3(0) \phi_4(2) \phi_{34}(02) \\
\quad + V_2(1,0) \phi_3(1) \phi_4(2) \phi_{34}(02) \\
\quad + V_2(0,1) \phi_3(1) \phi_4(2) \phi_{34}(12) \\
\quad m_{+++3} = V_2(1,1) \phi_3(1) \phi_4(3) \phi_{34}(13).
\end{array}
\end{align*}
\]

Application of Equation 9 and utilizing the result in (10) saves four operations.

Furthermore it is seen that the first and the second term in the calculation of \(m_{+++1}\) in Equation 10 are the same except for the \(V\) factor. The same is true for the second and the third term in \(m_{+++2}\). Therefore, using the distributive law of multiplication over summation, (10) can be rewritten as

\[
\begin{align*}
\quad m_{+++0} = V_2(0,0) \phi_3(0) \phi_4(0) \phi_{34}(00)
\end{align*}
\]
Estimating Quasi-Loglinear

12

\[ m_{+++1} = [V_2(1,0)+V_2(0,1)] \Phi_3(0) \Phi_4(1) \Phi_{34}(01) \]
\[ + V_2(0,0) \Phi_3(1) \Phi_4(1) \Phi_{34}(11) \]
\[ m_{+++2} = V_2(1,1) \Phi_3(0) \Phi_4(2) \Phi_{34}(02) \]
\[ + [V_2(1,0)+V_2(0,1)] \Phi_3(1) \Phi_4(2) \Phi_{34}(12) \]
\[ m_{+++3} = V_2(1,1) \Phi_3(1) \Phi_4(3) \Phi_{34}(13) \]

This reduces the number of operations required even further. Using (11) instead of (10) saves 2 x 3 multiplication operations in the calculation of \( m_{+++1} \) and \( m_{+++2} \). It is seen from (11) that the same sum of \( V \)-terms occurs in the calculation of both \( m_{+++1} \) and \( m_{+++2} \). Consequently, the number of summations can be reduced by one by first calculating all sums of \( V \)-terms and then applying the results in (11).

In the calculation of \( m_{+++1} \), the \( V \)-terms \( V_2(1,0) \) and \( V_2(0,1) \) which are to be summed have in common that the scores of the items on which they depend have the same sum score, i.e., \( 1 + 0 = 1 \) and \( 0 + 1 = 1 \). This is caused by the fact that \( x_3 = 0 \) in \( \Phi_3(x_3) \), \( t = 1 \) in \( m_{+++t} \) and \( t = x_1 + x_2 + x_3 \) so that \( x_1 + x_2 = t-x_3 = t_2 = 1 \).

Let \( S_2(t_2) \) be the sum over the first two variables of the \( V_2 \)-terms for which the partial sum score over the first two items is equal to \( t_2 \). The index of \( S \) denotes that the parameters that depend on the first two variables have been processed. This sum becomes

\[ S_2(t_2) = \sum_{x_1,x_2} V_2(x_1,x_2) \]

for \( x_1 = 0,1; x_2 = 0,1; t_2 = 0,1,2; t_2 = x_1 + x_2 \), where \( \sum_{(a,b)}^{(c)} \) means summation of the argument over the possible values of \( a \) and \( b \).
Estimating Quasi-Loglinear

13

b, holding c constant. In (12) this means that for each value of the sum $t_2$ of the first two items, the V-terms are summed over $x_1$ and $x_2$. Written in full, Equation 12 gives:

\begin{align}
S_2(0) &= V_2(0,0) \\
S_2(1) &= V_2(0,1) + V_2(1,0) \\
S_2(2) &= V_2(1,1)
\end{align}

It is seen that there is only one V-term for which the sum score $t_2$ of the first two items is zero. Therefore for $t_2 = 0$ the sum of V-terms over the items $x_1$ and $x_2$ is only one term: $V_2(0,0)$. In the second equation there are two V-terms for which the sum score $t_2$ of the first two items is one, $V_2(0,1)$ and $V_2(1,0)$. Therefore for $t_2 = 1$ the sum of the V-terms over $x_1$ and $x_2$ is the sum of $V_2(0,1)$ and $V_2(1,0)$. Note that since $x_1$ is equal to $t_2 - x_2$, summing over only $x_1$ for constant $t_2$ is the same as summing over both $x_1$ and $x_2$ for constant $t_2$. A general definition of the functions $S_i$ for $i = 1, \ldots, k$ is given later. The present definition of $S_2$ is consistent with that general definition. Substitution of (13) in (11) gives:

\begin{align}
m_{+++0} &= S_2(0) \phi_3(0) \phi_4(0) \phi_{34}(00) \\
m_{+++1} &= S_2(1) \phi_3(0) \phi_4(1) \phi_{34}(01) \\
&\quad + S_2(0) \phi_3(1) \phi_4(1) \phi_{34}(11) \\
m_{+++2} &= S_2(2) \phi_3(0) \phi_4(2) \phi_{34}(02) \\
&\quad + S_2(1) \phi_3(1) \phi_4(2) \phi_{34}(12) \\
m_{+++3} &= S_2(2) \phi_3(1) \phi_4(3) \phi_{34}(13)
\end{align}
In Equation 8 through 14 the calculations for the first two items have been performed and expressed in $S_2(t_2)$. The calculations involved a multiplication step (9) and a summation step (12). In the multiplication step, for each of the possible combinations of responses on the first two variables, the corresponding parameters that depend on these two variables are multiplied and assigned to a V-term. In the summation step these V-terms were summed over the values of the first and the second variable, where the partial sum score $t_2$ of the first two items was held constant. By holding the partial sum score constant, it is assured that the restriction $x_4 = x_1 + x_2 + x_3$ can be respected. Since the first two items are processed the restriction becomes

\begin{equation}
\text{(15)} \quad x_4 = t_2 + x_3.
\end{equation}

The generic form of Equation 14 becomes

\begin{equation}
\text{(16)} \quad m_{+++x_4} = \left( \begin{array}{c}
(t_3) \\
(t_2, x_3) \\
(t_3, x_4) \\
(t_2, x_3, x_4)
\end{array} \right) S_2(t_2) \Phi_3(x_3) \Phi_4(x_4) \Phi_{34}(x_3^x_4),
\end{equation}

for $t_2 = 0, 1, 2$; $x_3 = 0, 1$; $x_4 = t_3 = t_2 + x_3$.

In a similar way, the third item can be processed performing a multiplication step and a summation step. Finally the sum score variable can be processed by performing a multiplication step.

The next multiplication step involves the product of the S-term and the parameters that depend on item three (see Equation 16). Two parameters depend on item three, the main effect parameter $\Phi_3(x_3)$ and the interaction effect parameter $\Phi_{34}(x_3^x_4)$ for the combination of item three and the sum score. The product is then
Estimating Quasi-Loglinear

\[(17) \quad V_3(t_2, x_3, x_4) = S_2(t_2) \phi_3(x_3) \phi_{34}(x_3 x_4)\]

for \(t_2 = 0, 1, 2; x_3 = 0, 1; x_4 = t_2 + x_3.\)

Where the function \(V_3\) is defined as the product of the \(S_2\)-term and the model parameters not yet processed that depend the third variable \((x_3)\). The function \(V_3\) itself depends on all the variables (partial sum score, items, or the sum score) on which one or more of these parameters or the \(S_2\)-term depend. In the present model the parameters \(\phi_3(x_3)\) and \(\phi_{34}(x_3, x_4)\) depend on variable 3. These parameters and the \(S_2(t_2)\) term depend on the variables \(t_2, x_3,\) and \(x_4.\)

Substituting (17) in (14) yields:

\[
\begin{align*}
\quad m_{+++0} &= V_3(0, 0, 0) \phi_4(0) \\
\quad m_{+++1} &= V_3(1, 0, 1) \phi_4(1) \\
&\quad + V_3(0, 1, 1) \phi_4(1) \\
\quad m_{+++2} &= V_3(2, 0, 2) \phi_4(2) \\
&\quad + V_3(1, 1, 2) \phi_4(2) \\
\quad m_{+++3} &= V_3(2, 1, 3) \phi_4(3).
\end{align*}
\]

(18)

Applying the distributive law, once more we can rewrite (18) as:

\[
\begin{align*}
\quad m_{+++0} &= V_3(0, 0, 0) \phi_4(0) \\
\quad m_{+++1} &= [V_3(1, 0, 1) + V_3(0, 1, 1)] \phi_4(1) \\
\quad m_{+++2} &= [V_3(2, 0, 2) + V_3(1, 1, 2)] \phi_4(2) \\
\quad m_{+++3} &= V_3(2, 1, 3) \phi_4(3).
\end{align*}
\]

(19)

The \(V_3\)-terms depend on the scores of the first three items. They depend directly on \(x_3\) and indirectly on \(x_1\) and \(x_2\) via the partial
sum score $t_2$. Note that, due to the interaction parameter $\phi_{34}(x_3x_4)$, the $V$-terms in (19) also depend on sum score $x_4$. Obviously the values that this total sum score can assume are restricted by the value of the partial sum score on which the $V$-term depends, in this case $x_4 = t_3$. If there were more than three items, say $k$, the sum-score could assume values larger or equal to $t_3$, i.e. $t_3 \leq x_k \leq k$.

As before the summed $V_3$-terms in (19) have the same partial sum score for the first three items, because $t_2 + x_3 = t_3$ is the same in each term. The sums in (19) can be replaced by:

$$S_3(t_3, x_4) = \sum_{t_2, x_3} V_3(t_2, x_3, x_4),$$

for $t_2 = 0, 1, 2$; $x_3 = 0, 1$; $x_4 = t_3 = t_2 + x_3$. The index of $S_3$ denotes that the parameters that depend on the first three variables have been processed. For each $t_3$ this sum becomes

$$S_3(0, 0) = V_3(0, 0, 0)$$
$$S_3(1, 1) = V_3(1, 0, 1) + V_3(0, 1, 1)$$
$$S_3(2, 2) = V_3(2, 0, 2) + V_3(1, 1, 2)$$
$$S_3(3, 3) = V_3(2, 1, 3)$$

so that

$$m_{+++0} = S_3(0, 0) \phi_4(0)$$
$$m_{+++1} = S_3(1, 1) \phi_4(1)$$
$$m_{+++2} = S_3(2, 2) \phi_4(2)$$
$$m_{+++3} = S_3(3, 3) \phi_4(3).$$
The expected marginal sums can now be obtained by performing one more multiplication step for the sumscore variable,

\[ V_4(t_3, x_4) = S_3(t_3, x_4) \phi(x_4) \]

\[ t_3 = 0, 1, 2, 3; \quad x_4 = t_3, \]

where the function \( V_4 \) is defined as the product of the \( S_3 \)-term and the model parameters not yet processed. The function \( V_4 \) itself depends on all the variables (partial sum score, items, or the sum score) on which one or more of these parameters or the \( S_3 \) term depend. Substituting (23) in (22)

\[ m_{+++0} = V_4(0,0) \]
\[ m_{+++1} = V_4(1,1) \]
\[ m_{+++2} = V_4(2,2) \]
\[ m_{+++3} = V_4(3,3) \]

the expected marginal sums are obtained.

In this example, a small number of items and a simple model was chosen for the example. In practice there will usually be many more items and the model may contain interactions of all orders. In what follows, this marginalization by variable algorithm is described for arbitrary number of items and an arbitrary quasi-loglinear model.
The Marginalization-By-Variable Algorithm

In this section, the Marginalization-By-Variable (MBV) algorithm is described for an arbitrary model (2), where the model terms \( \Phi_{A_1}(x_{A_1}), \ldots, \Phi_{A_j}(x_{A_j}), \ldots, \Phi_{A_S}(x_{A_S}) \) are main and interaction effects of the variables whose indices are in the sets \( A_1, \ldots, A_j, \ldots, A_S \). In the model term \( \Phi_{A_j}(x_{A_j}) \), the vector \( x_{A_j} \) refers to a generic value of the vector of variables whose indices are in \( A_j \). Thus each model term corresponds to a set of parameters with specific values of \( x_{A_j} \). For convenience, we will refer to a model term \( \Phi_{A_j}(x_{A_j}) \) by the index \( j \) of the set \( A_j \) that characterizes that model term.

To estimate the model parameters using the iterative proportional fitting algorithm (5), expected sufficient marginal counts \( m_{x_{A_1}^{A_1}}, \ldots, m_{x_{A_j}^{A_j}}, \ldots, m_{x_{A_S}^{A_S}} \) must be calculated. The calculation of one expected marginal table \( m_{x_{A_j}^{A_j}} \) is described. To obtain this table, summations must be performed over all possible values of the remaining variable indices not in \( A_j \).

Although \( A_j \) may be every subset of variable indices, for simplicity of exposition, it is assumed that \( A_j \) contains the indices of the first \( v \) \((k)\) variables, i.e. \( \bar{A}_j = \{v+1, \ldots, k+1\} \). This presents no loss of generality since the original set of items can be renumbered arbitrarily to fit this representation. Moreover, all tables that do not depend on the sum-score variable can be obtained by summing \( m_{x_{A_j}^{A_j}} \) over the sum-score variable.

Just as in the example above, the expected marginal table \( m_{x_{A_j}^{A_j}} \) is obtained by repeated multiplications (c.f. (9) and (17)) and...
summing (c.f. (12) and (20)) of the parameters that depend on one
the variable at a time. The result is multiplied by the remaining
parameter (cf. (23)).

To obtain \( \mathbf{m}_{\mathbf{A}} \), \( 2v + 1 \) steps are needed in the Algorithm below:
2v steps for multiplication and summation of the parameters
corresponding to the first \( v \) items and one more step to multiply the
result with the remaining parameters to obtain the marginal table.
An optional summation has to be added if the marginal table may not
depend on the sum score.

Six steps are described: step 1, 2, 3, 4, \( 2i-1 \), \( 2i \), and
\( 2v+1 \). The odd numbered steps involve multiplication operations
while the even numbered steps involve summation operations. Step 1
and 3 correspond to multiplication of the model terms depending on
variables 1 and 2 respectively. The summation in step 2 does not
have any effect but is added for later reference. Therefore, the
result of step 1, 2, and 3 correspond to multiplication of the model
terms depending on variable 1 and 2. In the example above, these
first three steps are summarized in Formula (9).

**Step 1**

Multiply the parameters depending on variable \( x_1 \). Let \( L_1 \) be the
set of model terms \( j \in \{ I, \ldots, s \} \) depending on variable 1 and let
\( B_1 \) be the set of indices of the variables on which these model
terms depend. Then, the product is obtained:

\[
V_1(x_{B_1}) = \prod_{j \in L_1} \phi_{A_j}(x_{A_j}),
\]
for all possible values that the vector $x_{B_1}$ can assume. If $x_b (b \in B_1)$ is an item ($b \leq k$) it can assume values $x = 0, 1$. All combinations of item responses can occur. If $x_b$ is the sum-score variable ($b = k + 1$) then the values it can assume depend on the values of the item responses:

$$x_{k+1} = t(x_{B_1}), \ldots, k - (\#(B) - t(x_{B_1}))$$

where $t(x_{B_1})$ is the sum score of the items in the vector $x_{B_1}$ and the function $\#(X)$ yields the number of elements of a set $X$. The formula shows that the sum score variable cannot be smaller than the sum scores of the items of vector $x_B$. Also, it cannot be larger than the total number of items minus the number of wrong responses ($\#(B) - t(x_{B_1})$) of the items in the vector $x_{B_1}$.

Note that since the sets $A_j (j \in L_1)$ are subsets of $B_1$, the values of $x_{A_j} (j \in L_1)$ are known if the value $x_{B_1}$ is known.

**Step 2**

Sum the result of step 1 over variable $x_1$ holding the sum score $t_1$ constant, where $t_1$ was defined as $x_1 + \ldots + x_i$ (the sum of the first $i$ item responses). Since in this step $t_1$ and $x_1$ are still equal, in fact no summations are performed. However for later reference the following sum is defined

$$(26) \quad S_1(t_1, x_{C_1}) = \sum_{t_1} V_1(x_{B_1})$$

for all possible values of $t_1$ and $x_{C_1}$, where $t_1 = x_1$ and
Estimating Quasi-Loglinear

$C_1 = B_1 - \{1\}$ is the set of subscripts of the variables on which $S$ depends.

The sum score $t_1$ can assume values 0 and 1 since it is equal to $x_1$. The items in $x_{C_1}$ can each assume the values 0 and 1. If $x_{C_1}$ also contains the sum-score variable $x_{k+1}$, the latter assumes the values

$$x_{k+1} = (t_1 + t(x_{C_1})), \ldots, (t_1 + t(x_{C_1}) + (k - 1 - \#(C_1))),$$

that is, the total sum score $x_{k+1}$ cannot be smaller than the sum score $t_1$ of the first item plus the sum score of the items in vector $x_{C_1}$. Also it cannot be larger than this value plus the maximum score that the remaining items, not in $C_1$ nor used in $t_1$, can assume.

**Step 3**

Multiply the result of step 2 with the model terms that depend on variable $x_2$ but are not used in step one, i.e. that are not in $L_1$. Let $L_2$ be the set of these model terms ($j \in \{1, \ldots, s\}$) and let $B_2$ be the set of indices on which these parameters or the result $S_1$ of step 2 depend. Then this product is defined as

$$(27) \quad V_2(t_1, x_{B_2}) = S_1(t_1, x_{C_1}) \cap \prod_{j \in L_2} A_j(x_{A_j})$$

for all possible values of $t_1$ and $x_{B_2}$. Note that since the sets $A_j$ ($j \in L_2$) and $C_1$ are subsets of $B_2$, the values of $x_{A_j}$ ($j \in L_2$) and $x_{C_1}$ are known if the value $x_{B_1}$ is known. The values that $t_1$ and $x_{B_2}$
can assume are defined analogously to those in step 2 with \( C_1 \) replaced by \( B_2 \).

**Step 4**

Sum the result of step 3 over \( t_1 \) and \( x_2 \) holding the sum score \( t_2 = x_1 + x_2 = t_1 + x_2 \) constant. This sum is

\[
S_2(t_2, x_2) = \sum_{(t_1, x_2)} V_2(t_1, x_2)
\]

for all possible values of \( t_2 \) and \( x_2 \) where \( C_2 \subseteq B_2 - \{2\} \). The values that \( t_2 \) and \( x_2 \) can take are defined analogously to those of \( t_1 \) and \( x_1 \) in step 2 (see below).

Steps similar to step 3 and 4 are performed for the remaining variables \( x_3, \ldots, x_i, \ldots, x_v \) over which one has to sum. In general these steps are as follows.

**Step (2i-1)**

Multiply the result of the previous step with the model terms depending on variable \( i \) that are not used in the one of the previous steps, i.e., the model terms \( \{ j \in \{1, \ldots, s\} \} \) that are not in \( L_1, \ldots, L_{i-1} \). Let \( L_i \) be the set of these parameters and let \( B_i \) be the set of indices on which these parameters or the result of the previous step depend. This step yields the product.

\[
V_i(t_{i-1}, x_{B_i}) = S_{i-1}(t_{i-1}, x_{C_i}) \prod_{j \in L_i} \Phi_{x_j}(x_{A_j})
\]

for all possible values of \( t_{i-1} \) and \( x_{B_i} \), where
Estimating Quasi-Loglinear

23

$C_{i-1} = B_{i-1} - \{i\}$. The values that $t_{i-1}$ and $x_{B_{i}}$ can assume are
defined analogously to the values of $t_1$ and $x_{C_1}$ in step 21.

\section{Step 21}

Sum the result of the previous step over $t_{i-1}$ and $x_i$ holding
constant the sumscore $t_i = x_1 + \ldots + x_i = t_{i-1} + x_i$. This step
yields the sum

\begin{equation}
S_i(t_i, x_{C_i}) = \sum_{x_{C_i}} S_{i-1}(t_{i-1}, x_{C_i}) V_i(t_{i-1}, x_{B_i})
\end{equation}

for all possible values of $t_i$ and $x_{C_i}$, where $C_i = B_i - \{i\}$.

The sum score $t_i = x_1 + \ldots + x_i$ can assume values 0 through i
since each of the items $x_1$, $\ldots$, $x_i$ can take the values 0 or 1.
Also, each of the items in $x_{C_i}$ can assume the values 0 and 1. Note
that none of the items in $x_{C_i}$ is used to calculate $t_i$ since $x_{C_i}$ is
the vector of item responses on which $S_i$ depends after summation
over the first $i$ variables. If $x_{C_i}$ also contains the sum score
variable $x_{k+1}$, it assumes the values

\begin{equation}
x_{k+1} = (t_i + t(x_{C_i})), \ldots,
\end{equation}

\begin{equation}
(t_i + t(x_{C_i}) + (k - i - \#(C_i)),
\end{equation}

that is, the total sum score $x_{k+1}$ in $x_{C_i}$, cannot be smaller than
the sum score $t_i$ of the first $i$ items plus the sum score $t(x_{C_i})$ of
the items in the vector $x_{C_i}$. Also, it cannot become larger than
this value plus the maximum sum score that the remaining items,
i.e. the items not in $x_{C_1}$ nor used to calculate $t_i$, can attain.

Step 2v+1

To calculate the expected marginal sums $m_{A_v}$ in (6) that depend on the variables $A_v = \{(v+1),\ldots,(k+1)\}$, the $S_v$-term resulting from the previous step is multiplied by the model terms that have not been used in the previous steps, i.e. the model terms that are not in $L_1,\ldots,L_v$. Let $L_{v+1}$ be the set of these model terms $(j)$, then the expected marginal sums are

$$(32) \quad m_{A_v} = S_v(t_v, x_v) \prod_{j \in L_{v+1}} \phi_{A_j}(x_{A_j})$$

for all possible values of $x_{A_v}$. Note that the sets $A_j (j \in L_{v+1})$ correspond to model terms that have not been used earlier. Consequently, they cannot contain any of the variables $1,\ldots,v$. Therefore, $A_j (j \in L_{v+1})$ are subsets of $A_v = \{v+1,\ldots,k+1\}$. Similarly since $S_v$ is obtained by summation over the variables $1,\ldots,v$ the result cannot depend on $x_1,\ldots,x_v$. Therefore $C_v$ is also a subset of $A_v = \{v+1,\ldots,k+1\}$.

Furthermore, since the model contains a main effect for each variable, each variable index $(v+1),\ldots,(k+1)$ occurs at least in one of the sets $A_j (j \in L_{v+1})$. Because of this and the observation above we have

$$A_v = \bigcup_{j \in L_{v+1}} A_j = \{v+1,\ldots,k+1\}$$

The variables in (32) are related in the following way, since the
sets \( C_v \) and \( A_j (j \in L_{v+1}) \) are subsets of \( A \), the values of \( \chi_{A_j} (j \in L_{v+1}) \) and \( \chi_{C_v} \) are contained in \( \chi_{A_j} \). Furthermore, the sum score variable \( \chi_{k+1} \) in the vector \( \chi_{A_j} \) is related to \( t_v \) and the values \( \chi_{A_j} \) for each of the model terms \( j \in L_{v+1} \) by the relation:

\[
t_{k+1} = t_v + \chi_{v+1} + \ldots + \chi_k,
\]

where \( \chi_{v+1}, \ldots, \chi_k \) can each take the values 0 or 1.

**Numbers of Calculations in the MBV-algorithm**

The MBV-algorithm was developed to reduce the number of operations necessary to calculate the expected sufficient marginal tables that must be used in the iterative proportional fitting algorithm (5). We will compare the number of calculations using the MBV-algorithm with the number of calculations that would have been necessary if all the expected cell frequencies had been summed to obtain the expected sufficient marginals. The comparison is done only for the quasi-independence model, i.e., the model containing main effect parameters only. Furthermore it is assumed that all input variables are dichotomously scored items.

Summing over all cells requires calculation of each expected cell frequency. One expected cell frequency involves the multiplication of the general mean parameter with \( k \) item parameters and one sum score parameter, i.e. \( k+1 \) multiplications. There are a total of \( 2^k \) cells in the table, hence

\[
2^k
\]
Estimating Quasi-Loglinear

26

\[(33) \quad (k + 1)2^k\]

multiplications have to be performed.

The expected marginal table for the responses on one item has two possible cells, so

\[(34) \quad 2^k - 2\]

summations are necessary to obtain this table from the \(2^k\) cell counts. Likewise, for the sum score marginal table, this number is

\[(35) \quad 2^k - (k + 1)\]

since the sumscore has \(k + 1\) possible values. In Table 1 the numbers of multiplications and summations are given for tests of different lengths. Obviously, if the number of items is large, this method is not feasible.

In the MBV algorithm the number of calculations to obtain the sumscore marginal is as follows. In the multiplication step (29) each of the specific values of elements of the codomain of \(V_i\) is obtained by multiplying an element of the \(S\)-term with each of the parameters whose index is in \(L_i\). If there are only main effects in the model, \(L_i\) has only one element: The number of the main effect term for variable \(i\). So only one multiplication is needed for each element of \(V_i\). The number of elements in \(V_i\) is equal to the product of the number of values that the partial sumscore \(t_{i-1}\) can assume and the number of values that the item responses \(x_{B_i}\) can assume.
Because we consider a main effects model $B_1 = \{1\}$, and hence $x_{B_1}$ is equal to $(x_1)$. The number of possible values of $t_{i-1}$ is 1 and the number of values of $x_1$ is 2. Consequently, one multiplication step involves $2i$ multiplications.

In the summation step, the $S$-term is obtained by summing certain elements of $V_i$ over $x_1$ and $t_{i-1}$. In this case $S_i$ depends on $t_i$ which can assume $i + 1$ values. Therefore, to obtain $S$ from $V$, $(2i) - (i + 1) = i - 1$ summations are necessary.

To obtain the marginal table for the sumscore variable $a$, a multiplication and a summation step must be performed for each of the $k$ items. Note that in the first step there is no $S$-term and only one (main effect) parameter. Hence, this step does not involve multiplication. Therefore, multiplication starts at the third step. In addition, each of the $(k + 1)$ values of the last $S$ term has to be multiplied with the sumscore parameter and the grand mean parameter. The number of the multiplications is therefore

$$\sum_{i=2}^{k} 2i + 2(k + 1) = \sum_{i=1}^{k} (2i + 2k)$$

$$= k(k + 1) + 2k$$

$$= k(k + 3)$$

There are summation steps for each of the $k$ items, therefore the number of summations is
Estimating Quasi-Loglinear

It can be seen from Table 1 that the number of multiplications and summations in the MBV algorithm remains within reasonable limits and is less than in the case of summing over all cells.

For an arbitrary model \( A_1, \ldots, A_s \) for \( k \) dichotomous items, the number of multiplications and summations are more difficult to calculate. Suppose again that marginalization has to be performed over the all observed variables. One multiplication step involves

\[
(38) \quad a_i \#(L_i),
\]

\( i = 2, \ldots, k \), multiplications where

\[
(39) \quad a_i = \#(\{V_i(\{t_i-1, x_{0_d_i}\})\})
\]

is the number of elements of \( V_i \) and \( \#(L_i) \) is the number of parameters to be multiplied with the preceding sum for each element of \( V_i \).

The first step does not involve an \( S \) term so that it involves \( a_1(\#L_1 - 1) \) multiplications. After \( k \) steps the \( S \) term depends only on \( x_4 \) and the marginal table can be obtained by multiplying each of its \( k + 1 \) elements with the sum score parameter and the grand mean, which gives \( 2(k + 1) \) multiplications. The total number of multiplications then becomes

\[
\sum_{j=1}^{k} (j - 1) = \frac{(k + 1)k}{2} - k
\]

\[
= \frac{k(k - 1)}{2}
\]
Estimating Quasi-Loglinear

(40) \[ \sum_{i=1}^{k} \#(L_i) a_i - a_1 + 2(k + 1) \]

The number of summations in each step is

(41) \[ a_i - b_i \]

where

(42) \[ b_i = \#([S_i(t_i,x_{c_i})]) \]

i.e. the number of specific values of \( S_i \). The total number of summations becomes

(43) \[ \sum_{i=1}^{k} (a_i - b_i) \]

The numbers \( a_i \) and \( b_i \) depend on the specification of the model. The number \( a_i \) of elements of \( V_i(t_{i-1},x_{B_i}) \) is the product of the number of partial sumscores \( t_{i-1} \) and the number of values of \( x_{B_i} \). The number of partial sumscores \( t_{i-1} \) is equal to \( i \). The number of values of \( x_{B_i} \) is the product of the number of values that the items whose numbers are in \( B_i \) can jointly assume and the number of values that the sumscores can assume, if its number is in \( B_i \).

The items whose numbers are in \( B_i \) can jointly assume

(44) \[ \#(B_i - \{k+1\}) / 2 \]
values. If the number of the sum score variable is present in $B_i$, i.e. $\#(\{k+1\} \cap D_i) = 1$, the number of values that the sum score variable can assume is equal to

\[(45) \quad 1 + k - \#(B_i).\]

Thus, the number elements of $V_i$ becomes

\[(46) \quad a_i = 2^{\#(B_i - \{k + 1\})} \cdot \frac{\#(\{k + 1\} \cap B_i)}{(1 + k - \#(B_i))}.
\]

The S-term is obtained from the V-term by summing the elements with the same partial sum score $t_i$ over the two values of item $i$. Consequently the number of values that the partial sum score can assume becomes $i + 1$ rather than $i$ and the number of values that $x_{C_i}$ can assume is half the number of values of $x_{B_i}$, hence

\[b_i = \frac{(i + 1)}{2} a_i,\]

The number of summations using the MBV algorithm thus becomes

\[(47) \quad \sum_{i=1}^{k} a_i \left(1 - \frac{(i + 1)}{2i}\right) = \sum_{i=1}^{k} a_i \frac{1}{2} (1 - \frac{1}{i}).\]

In contrast, if the marginal table for the sum score variable is calculated by summing all cell frequencies, the number of summations is equal to the number of cells of the full table minus the number of cells of the marginal table.
Estimating Quasi-Loglinear

\[ (48) \quad 2^k - (k + 1), \]

and the number of multiplications is equal to

\[ (49) \quad 2^k(s - 1), \]

which is the product of the number of cells of the full table and the number of parameters minus one.

It is seen from Formula (46) and (47) that the number of summations depends exponentially on the number of variables in \( B_i \). This number, in turn, depends on the number of variables that variable \( i \) interacts with. The number of summations (48) in the summing over all cells algorithm, however, will always depend exponentially on the number of items \( k \).

Comparing the number of summations (47) in the MBV algorithm with the number of summations (48) needed when summing over all cells, it can be shown that the former is smaller or equal to the latter. Equality occurs if the model is the saturated model (see Appendix I).

Application of the MBV Algorithm

To estimate quasi-loglinear models for the Rasch model when the number of items is large, the computer program GELORA (Generalized Loglinear Rasch Modelling) was written. GELORA is a Pascal program that calculates the parameter estimates using the methods described above.
To evaluate the applicability of the algorithms, test data conforming to the Rasch model were generated for 20 items. The item difficulties were randomly chosen from the uniform distribution over the interval [-2,2]. Latent trait values for 10,000 cases were drawn from a uniform distribution over the [-3,3] interval.

Loglinear Rasch models were then fitted to these data. Seventeen computer runs were made for different subsets of items, where the first subset contained the first four items, the second subset contained the first six items etcetera.

In Table 2 the numbers of iterations, the mean CPU time per iteration, the CPU time needed for input and initialization, and the total CPU time of the problem run are shown. Iterations were performed until none of the parameter estimates could be improved by more than .005. A VAX 8750 computer was used. From Table 2 it is seen that the number of iterations and the mean CPU times per iteration do not increase dramatically compared to the number of items in the test.

In Table 3 the real item difficulties and the estimated item difficulties values of all 20 items are given. The item parameter estimates were obtained by the GELORA program and by the PML (Gustaffson, 1977) program. PML calculates the CML estimates of the item parameters with Andersen's (1972) method. In both cases the first item difficulty parameter was set equal to its real value. Furthermore, the iterations were stopped until none of the parameter estimates could be improved by more than .0001. It can be seen from Table 3 that both solutions are identical up to the second decimal place.
The algorithm was also used with simulated Rasch data for 40 items. With 40 items the solution was reached after 29 iterations. Each iteration took approximately 155 CPU seconds. This shows that maximum likelihood estimates in quasi-loglinear Rasch models can be obtained for practical numbers of items.

Conclusion

In this paper an algorithm is presented that calculates the parameter of quasi-loglinear models for the Rasch table from the expected sufficient statistics by an efficient method. The method is implemented in the program GELORA. The program facilitates the application of Rasch item analysis by quasi-loglinear models.
In the case of the saturated model $D_1 = \{1, \ldots, k+1\}$ so that

Formula $46$:

$$a_1 = i_2(k - 1 + 1)$$

The number of summations, Formula (47), is:

$$\sum_{i=1}^{k} i_2 k - i (1 - \frac{1}{i})$$

**THEOREM**

For all integers $k \geq 1$,

$$\sum_{i=1}^{k} i_2 k - i (1 - \frac{1}{i}) = 2^k - (k + 1).$$

**Proof (by induction):** For $k = 1$, the theorem is true because both sides reduce to zero.

The induction hypothesis is that for $r \geq 1$

$$\sum_{i=1}^{r} i_2 r - i (1 - \frac{1}{i}) = 2^r - (r + 1).$$

Hence

$$\sum_{i=1}^{r+1} i_2 (r + 1) - i (1 - \frac{1}{i})$$

$$= \sum_{i=1}^{r} i_2 r - i (1 - \frac{1}{i}) + i_2 (r + 1) - (r + 1)$$

$$= 2^r - (r + 1) + i_2 (r + 1) - (r + 1)$$

$$= 2^r - (r + 1) + (r + 1) - 1$$

$$= 2^{r+1} - (r + 2).$$
\[ \begin{align*}
&= \sum_{i=1}^{r+1} 2^r - i(1 - \frac{1}{i}) \\
&= \sum_{i=1}^{r} [2^r - (1 - \frac{1}{i})] + 2(r + 1) 2^{-1} (1 - \frac{1}{r+1}) \\
&= 2(2^r - (r + 1)) + r = 2^r + 1 - ((r + 1) + 1)
\]

which proves the theorem for \( k = r + 1 \).
Table 1

Numbers of Multiplications and Summations Required by Summing over all cells and the MBV Method to Calculate the Sumscore Marginal

<table>
<thead>
<tr>
<th>Number of Items</th>
<th>Summing all cells</th>
<th>MBV Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>+</td>
</tr>
<tr>
<td>5</td>
<td>192</td>
<td>26</td>
</tr>
<tr>
<td>6</td>
<td>448</td>
<td>57</td>
</tr>
<tr>
<td>7</td>
<td>1024</td>
<td>120</td>
</tr>
<tr>
<td>8</td>
<td>2304</td>
<td>247</td>
</tr>
<tr>
<td>9</td>
<td>5120</td>
<td>502</td>
</tr>
<tr>
<td>10</td>
<td>11264</td>
<td>1013</td>
</tr>
<tr>
<td>11</td>
<td>24576</td>
<td>2036</td>
</tr>
<tr>
<td>12</td>
<td>53248</td>
<td>4083</td>
</tr>
<tr>
<td>13</td>
<td>114688</td>
<td>8178</td>
</tr>
<tr>
<td>14</td>
<td>245760</td>
<td>16369</td>
</tr>
<tr>
<td>15</td>
<td>524288</td>
<td>32752</td>
</tr>
<tr>
<td>16</td>
<td>1114112</td>
<td>65519</td>
</tr>
<tr>
<td>17</td>
<td>2359296</td>
<td>131054</td>
</tr>
<tr>
<td>18</td>
<td>4980736</td>
<td>262125</td>
</tr>
<tr>
<td>19</td>
<td>10485760</td>
<td>524268</td>
</tr>
<tr>
<td>20</td>
<td>22020096</td>
<td>1048555</td>
</tr>
</tbody>
</table>
Table 2

Numbers of Iterations and Mean CPU Times for in Estimating Rasch models with GELORA.

<table>
<thead>
<tr>
<th>Number of Items</th>
<th>Number of Iterations</th>
<th>CPU Time</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Per Iteration</td>
<td>Input and Initialisation</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>0.3</td>
<td>10.0</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>0.7</td>
<td>14.3</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>1.5</td>
<td>18.5</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>2.6</td>
<td>24.7</td>
</tr>
<tr>
<td>12</td>
<td>10</td>
<td>4.3</td>
<td>32.4</td>
</tr>
<tr>
<td>14</td>
<td>11</td>
<td>6.6</td>
<td>39.0</td>
</tr>
<tr>
<td>16</td>
<td>11</td>
<td>9.5</td>
<td>48.2</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>14.2</td>
<td>58.0</td>
</tr>
<tr>
<td>20</td>
<td>13</td>
<td>18.8</td>
<td>68.7</td>
</tr>
</tbody>
</table>
Table 3

Real and Estimated item Difficulties for Simulated Data

| Item   | 1    | 3    | 4    | 5    | 6    | 7    | 8    | 9    | 10   | 11   | 12   | 13   | 14   | 15   | 16   | 17   | 18   | 19   | 20   |
|--------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Real   |      |      |      |      | 1.354| 1.690| 0.577| -1.270| -0.155|      |      |      |      |      |      |      |      |      |
| GELORA |      |      |      |      | 1.318| 1.636| 0.618| -1.350| -0.154|      |      |      |      |      |      |      |      |      |
| PML    |      |      |      |      | 1.318| 1.636| 0.618| -1.349| -0.153|      |      |      |      |      |      |      |      |      |
| Real   |      |      |      | 1.302| 1.352| -0.823| -0.883| -1.754|      |      |      |      |      |      |      |      |      |      |
| GELORA |      |      |      | 1.243| 1.282| -0.858| 0.871| -1.801|      |      |      |      |      |      |      |      |      |      |
| PML    |      |      |      | 1.244| 1.284| -0.857| 0.871| -1.801|      |      |      |      |      |      |      |      |      |      |
| Real   |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| GELORA |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| PML    |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |

*) The estimated parameter of the first item was set equal to the real parameter to fix the scale
References


Estimating Quasi-Loglinear


multivariate axial symmetry. *Applied Psychological Measurement*,
5, 263-269.

models in the assessment of mastery. *Journal of Educational
Statistics*, 2, 99-120.


Rasch, G. (1960). *Probabilistic models for some intelligence and
attainment tests*. Copenhagen: Peadagogiske Institut.

comparisons, In: L.T.J. van der Kamp & C.A.J. Vlek (Eos.),

Tjur, T. (1982). A connection between Rasch's item analysis model
and a multiplicative Poisson model. *Scandinavian Journal of

van der Linden, W.J. (1980). Simple estimators for the simple
latent class mastery testing model. Paper presented at the
International Symposium on Educational Testing, Antwerp, Belgium,
June 24-27.

Vijn, P. & Mellenberg, G.J. (October, 1982). The Rasch model versus
the loglinear model. [Révész Berichten no. 42], *Psychologisch
Laboratorium, Universiteit van Amsterdam.*

Wright, B.D. & Panchapakesan, N. (1969). A procedure for sample-
free item analysis. *Educational and Psychological Measurement*,
Titles of Recent Research Reports

RR-86-1  W.J. van der Linden, The use of test scores for classification decisions with threshold utility

RR-86-2  H. Kelderman, Item bias detection using the loglinear Rasch model: Observed and unobserved subgroups

RR-86-3  E. Boekkooi-Timminga, Simultaneous test construction by zero-one programming

RR-86-4  W.J. van der Linden, & E. Boekkooi-Timminga, A zero-one programming approach to Gulliksen's matched random subtests method

RR-86-5  E. van der Burg, J. de Leeuw, & R. Verdegaal, Homogeneity analysis with k sets of variables: An alternating least squares method with optimal scaling features

RR-86-6  W.J. van der Linden, & T.J.H.M. Eggen, An empirical Bayes approach to item banking

RR-86-7  E. Boekkooi-Timminga, Algorithms for the construction of parallel tests by zero-one programming

RR-86-8  T.J.H.M. Eggen, & W.J. van der Linden, The use of models for paired comparisons with ties

RR-86-9  H. Kelderman, Common item equating using the loglinear Rasch model
RR-86-10 W.J. van der Linden, & M.A. Zwarts, Some procedures for computerized ability testing

RR-87-1 R. Engelen, Semiparametric estimation in the Rasch model

RR-87-2 W.J. van der Linden (Ed.), IRT-based test construction

RR-87-3 R. Engelen, P. Thommassen, & W. Vervaat, Ignatov’s theorem: A new and short proof

RR-87-4 E. van der Burg, & J. de Leeuw, Use of the multinomial jackknife and bootstrap in generalized nonlinear canonical correlation analysis

RR-87-5 H. Kelderman, Estimating quasi-loglinear models for a Rasch table if the numbers of items is large

Research Reports can be obtained at costs from Mediatheek, Faculteit Toegepaste Onderwijskunde, Universiteit Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.
A publication by the Department of Education of the University of Twente