An IRT-based Parameterization for Conditional Probability Tables

Russell Almond
Florida State University
Educational Testing Models

- Usually some collection of latent variables representing target of assessment: \textit{proficiency model, } \theta_i
  - Usually positively correlated
- Each item (task) presented to examinee has one or more observable outcome variables \(X_{ij}\)
- \textit{Evidence model for Task } j \textit{ is } Pr(X_{ij} | \theta_i)
Monotonic and Parametric CPTs

• Both parent (proficiency) and child (observable) variables are ordered categorical

• Want CPTs to be monotonically increasing: i.e., higher values of proficiency imply higher probability of better outcomes

• Because proficiency variables are correlated, may only be a few observations for some rows of CPTs
Discrete IRT (2PL) model

• Imagine a case with a single parent and a binary (correct/incorrect) child.

• Map states of parent variable onto a continuous scale: effective theta, \( \tilde{\theta}_m \)

• Plug into IRT equation to get conditional probability of “correct”

\[
\Pr(Y_j = 1|X = m) = \logit^{-1}
\left[
1.7 a_j (\tilde{\theta}_m - b_j)
\right]
\]

• \( a_j \) – discrimination parameter
• \( b_j \) – difficulty parameter
• 1.7 – Scaling constant (makes logistic curve look like normal ogive)
Multivariate Models: Combination Rules

• For Multiple Parents, assign each parent $j$ an effective theta at each level $k$, $\theta_{j,k}$

• Combine Using a Combination Rule (Structure Function)

$$s\left(\tilde{\theta}_{1,k_1}, \ldots, \tilde{\theta}_{J,k_J}\right)$$

• Possible Structure Functions:
  • Compensatory = average
  • Conjunctive = min
  • Disjunctive = max
  • Inhibitor: e.g. level $k^*$ on : $\theta_1 \begin{cases} 
  s(\tilde{\theta}_{1,k_1}, \ldots, \tilde{\theta}_{J,k_J}) & \text{if } k_1 > k^* \\
  \tilde{\theta}_0 & \text{if } k_1 \leq k^*
  \end{cases}$

  where $\theta_0$ is some low value.
DiBello--Samejima Models

• Single parent version
• Map each level of parent state to “effective theta” on IRT (N(0,1)) scale, $\tilde{\theta}_k$
• Now plug into Samejima graded response model to get probability of outcome
• Uses standard IRT parameters, “difficulty” and “discrimination”
• DiBello--Normal model uses regression model rather than graded response
Samejima’s Graded Response Model

Samejima’s (1969) psychometric model for graded responses:

\[
\Pr(X_{i,j} \geq k \mid \theta_i) = \logit^{-1}(a_j \theta_i + b_{j,k})
\]

\[
\Pr(X_{i,j} = k \mid \theta_i) = \Pr(X_{i,j} \geq k \mid \theta_i) - \Pr(X_{i,j} \geq k - 1 \mid \theta_i)
\]
The “Effective $\theta$” Method (2): Conditional Probabilities for Three $\theta$’s

<table>
<thead>
<tr>
<th>$\theta$</th>
<th>$X=1$ (Poor)</th>
<th>$X=2$ (Okay)</th>
<th>$X=3$ (Good)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low= -1.8</td>
<td>.70</td>
<td>.25</td>
<td>.05</td>
</tr>
<tr>
<td>Med= -.4</td>
<td>.35</td>
<td>.40</td>
<td>.25</td>
</tr>
<tr>
<td>High= 1.0</td>
<td>.10</td>
<td>.40</td>
<td>.50</td>
</tr>
</tbody>
</table>
Example (Biomass)

DKMendel is the student-model variable that determines probabilities of response to the several observable variables in the Mode of Inheritance chart.

Context is a parent that induces conditional dependence among these observations, for reasons other than the DKMendel (e.g., did not understand what was required in task).
Effective Thetas for Compensatory Relationship

\[ \tilde{\theta}_{j,k} \] equally spaced normal quantiles

\[ a_{S_1} = 1 \quad a_{\text{Context}} = .75 \quad b_j = -1 \]

<table>
<thead>
<tr>
<th>S1</th>
<th>Context</th>
<th>S1.theta</th>
<th>Context.theta</th>
<th>Effective.theta</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Familiar</td>
<td>0.97</td>
<td>0.67</td>
<td>2.04</td>
</tr>
<tr>
<td>Medium</td>
<td>Familiar</td>
<td>0.00</td>
<td>0.67</td>
<td>1.36</td>
</tr>
<tr>
<td>Low</td>
<td>Familiar</td>
<td>-0.97</td>
<td>0.67</td>
<td>0.67</td>
</tr>
<tr>
<td>High</td>
<td>Unfamiliar</td>
<td>0.97</td>
<td>-0.67</td>
<td>1.33</td>
</tr>
<tr>
<td>Medium</td>
<td>Unfamiliar</td>
<td>0.00</td>
<td>-0.67</td>
<td>0.64</td>
</tr>
<tr>
<td>Low</td>
<td>Unfamiliar</td>
<td>-0.97</td>
<td>-0.67</td>
<td>-0.04</td>
</tr>
</tbody>
</table>
Effective Theta to CPT

Introduce new parameter $d_{inc}$ as spread between difficulties in Samejima model

$$b_{i,Full} = b_j + d_{inc}/2 \quad \quad b_{j,Partial} = b_j - d_{inc}/2$$

Conditional probability table for $d_{inc} = 1$ is then...

<table>
<thead>
<tr>
<th>S1</th>
<th>Context</th>
<th>Effective.theta</th>
<th>Full</th>
<th>Partial</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Familiar</td>
<td>2.04</td>
<td>0.73</td>
<td>0.15</td>
<td>0.12</td>
</tr>
<tr>
<td>Medium</td>
<td>Familiar</td>
<td>1.36</td>
<td>0.62</td>
<td>0.20</td>
<td>0.18</td>
</tr>
<tr>
<td>Low</td>
<td>Familiar</td>
<td>0.67</td>
<td>0.39</td>
<td>0.24</td>
<td>0.37</td>
</tr>
<tr>
<td>High</td>
<td>Unfamiliar</td>
<td>1.33</td>
<td>0.50</td>
<td>0.23</td>
<td>0.27</td>
</tr>
<tr>
<td>Medium</td>
<td>Unfamiliar</td>
<td>0.64</td>
<td>0.50</td>
<td>0.23</td>
<td>0.27</td>
</tr>
<tr>
<td>Low</td>
<td>Unfamiliar</td>
<td>−0.04</td>
<td>0.39</td>
<td>0.24</td>
<td>0.37</td>
</tr>
</tbody>
</table>
Limitation of Graded Response

• Samejima's graded response model requires curves to be parallel.
• Slope parameters must be the same (intercepts increasing)
• Combination functions must be the same
Generalized Partial Credit Model

- Muraki (1992)
- Focuses on state transitions
  \[ \Pr(X \geq m + 1 | X \geq m) \]
- Can use different slopes, sets of parents and combination rules for different state transitions
- Graded Response, and Partial Credit are examples of link functions that go from linear predictor to probabilities
CPTtools framework

Building a CPT requires three steps:

1. Map each parent state into a *effective theta* for that parent

2. Combine the parent effective thetas to an effective theta for each row of the CPT using one (or more) *combination rules*
   - Combination rules generally take one or more (often one for each parent variable) *discrimination parameters* which weight the parent variable contributions (log alphas)
   - Combination rules generally take one or more *difficulty parameters* (often one for each state of the child variable) which shift the average probability of a correct response (betas)

3. Map the effect theta for each row into a conditional probability of seeing each state using a *link function*
   - Link functions can take a scaling parameter. (link scale)
Parent level effective thetas

- Effective theta scale is a logit scale corresponds to mean 0 SD 1 in a “standard” population.
- Want the effective theta values to be equally spaced on this scale
- Want the marginal distribution implied by the effective thetas to be uniform (unit of the combination operator)
- What the effective theta transformation to be effectively invertible (this is reason to add the 1.7 to the IRT equation).
Equally spaced quantiles of the normal distribution

• Suppose variable has $M$ states: $0, \ldots, M-1$
• Want the midpoint of the interval going from probability $m/M$ to $(m+1)/M$.
• Solution is to map state $m$ onto $\Phi^{-1} \left( \frac{m + 1/2}{M} \right)$

• R code: `qnorm((1:M) - .5)/M)`
Combination Rules

• Compensatory – more of one skill compensates for lack of another
• Conjunctive – weakest skill dominates relationship
• Disjunctive – strongest skill dominates relationship
• Inhibitor – minimum threshold of Skill 1 needed, then Skill 2 takes over (special case of conjunctive)
• Offset Conjunctive – like conjunctive model, but with separate $b$’s for each parent instead of separate $a$’s
• Offset Disjunctive – like disjunctive rule, but with separate $b$’s for each parent instead of separate $a$’s.
Compensatory Rule

• Weighted average of inputs
• Weights are (as we often want the weights to be positive we often use as the parameter).

\[ \tilde{\theta} = \frac{1}{\sqrt{K}} \sum \alpha_{k,s} \theta_{km_k} - \beta_{js} \]

• \(s\) is state of child variable
• Factor \(1/\sqrt{K}\) is a variance stabilization term (makes variance stay the same as number of parents changes)
Conjunctive and Disjunctive rules

• Same setup, except replace sum with max and variance stabilization term is no longer needed:
  • Conjunctive: \( \tilde{\theta} = \min \alpha_{ks} \theta_{km_k} - \beta_{js} \)
  • Disjunctive: \( \tilde{\theta} = \max \alpha_{ks} \theta_{km_k} - \beta_{js} \)
  • Inhibitor:

\[
\tilde{\theta} = \begin{cases} 
\alpha_{2s} \theta_{2m_2} - \beta_{js}, & m_1 > m^*_1 \\
\alpha_{2s} \theta_{2,0} - \beta_{js}, & \text{otherwise}
\end{cases}
\]
Offset Conjunctive and Disjunctive

• Separate slopes doesn’t really make sense for conjunctive and disjunctive models
• Separate intercepts, i.e., a different difficulty for each parent variable, does.
• Multiple betas, one alpha
• Conjunctive: \[ \tilde{\theta} = \alpha_{js} \min(\theta_{km_k} - \beta_{ks}) \]
• Disjunctive: \[ \tilde{\theta} = \alpha_{js} \max(\theta_{km_k} - \beta_{ks}) \]
Link functions

• Graded Response model
  • Models for each value of s
  • In order to keep the curves from crossing, discrimination parameters must be the same for all s

• Normal (Regression) model
  • Effective theta is mean predictor
  • Add a residual variance (link scale parameter)
  • Calculate probabilities that value falls into certain regions

• Generalized partial credit model
  • Models state transitions
  • Does not need the discrimination parameters to be the same
  • Does not even need the combination rules to be the same
Normal Link function

• As with effective theta transformation, start by dividing theta region up into intervals
  • Equally spaced
  • Spaced to achieve a certain marginal distribution for Y
• Calculate offset curve:
  • mean is effective theta
  • SD, $\sigma$, is link scale parameter
• Conditional probabilities:
  • area under curve between cut points
Conjunctive-Normal model

• This is essentially a regression

\[ R^2 = \frac{\sum_{k=1}^{K} \alpha_k^2 / K}{\sigma^2 + \sum_{k=1}^{K} \alpha_k^2 / K} \]

• Note: If child value is a proficiency variable, this is a latent variable regression. Correlation should be higher than you think.
Generalized Partial Credit Link

• Set up a series of conditional probabilities:
  \[ P_{js|s-1}(\tilde{\theta}_i) = \Pr(Y_{ij} \geq s|Y_{ij} \geq s-1, \tilde{\theta}_i) = \logit^{-1}(1.7Z_{js}(\tilde{\theta}_i)) \]

• Probability of \( Y \) being in State \( s \) is:
  \[
  \Pr(V_{ij} = s|\tilde{\theta}_i) = \frac{\prod_{r=0}^{s} P_{jr|r-1}(\tilde{\theta}_i)}{C},
  \]
  where \( C \) is a normalization constant.

• Can convert the products to sums:
  \[
  \Pr(V_{ij} = s|\tilde{\theta}_i) = \frac{\exp(1.7 \sum_{r=0}^{s} Z_{jr}(\tilde{\theta}_i))}{\sum_{r=0}^{S_j} \exp(1.7 \sum_{r=0}^{R} Z_{jr}(\tilde{\theta}_i))}.
  \]
Discrete Partial Credit Model (DPC)

- $Z()$ is the combination rule (structure function)
- $Z_{jr}()$ describes how skills combine to make transition between state $r-1$ and $r$.
- $Z_{j0}() = 0$
- Although functional form is commonly taken as the same for all states, it does not need to be!
- This allows us to model different cognitive processes at different steps
Example: Math Word Problem

• Based on unpublished analysis by Cocke and Guo (personal communication 2011-07-11)

• Next Generation Sunshine State Standards Benchmark, MA.6.A.5.1, “Use equivalent forms of fractions, decimals, and percents to solve problems” (NGSSSS, 2013)

• Sample problem:
  John scored 75% on a test and Mary has 8 out of 12 correct on the same test. Each test item is worth the same amount of points. Who has the better score?
# Scoring Rubric

<table>
<thead>
<tr>
<th>Score Point</th>
<th>Description</th>
<th>Skills Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Null response or off track</td>
<td>None</td>
</tr>
<tr>
<td>1</td>
<td>Recognizes 75% and 8/12 as key elements</td>
<td>Mathematical Language</td>
</tr>
<tr>
<td>2</td>
<td>Converts two fractions to a common form</td>
<td>Convert Fractions</td>
</tr>
<tr>
<td>3</td>
<td>Makes the correct comparison</td>
<td>Compare Fraction &amp; Mathematical Language</td>
</tr>
</tbody>
</table>
Model Refinement

• Collapse categories 2 and 3 as very few 2’s observed in practice

• Combine *Convert fractions* and *Compare fractions* into *Fraction manipulation*

• Need two combination rules:
  • 0 → 1: Only one skill relevant. Can use any rule, choose compensatory because it is easiest to work with. Do selection by setting discrimination for *fraction manipulation* to 0.
  • 1 → 2: Both skills necessary, but inhibitor model: only a minimal level of mathematical language is necessary.
# Effective Thetas and Z’s

<table>
<thead>
<tr>
<th>Mathematical Language</th>
<th>$\theta_{i'1}$</th>
<th>Manipulate Fractions</th>
<th>$\theta_{i'2}$</th>
<th>$Z_{j0}(\theta_{i'})$</th>
<th>$Z_{j1}(\theta_{i'})$</th>
<th>$Z_{j2}(\theta_{i'})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>high</td>
<td>+0.97</td>
<td>high</td>
<td>+0.97</td>
<td>0</td>
<td>+1.47</td>
<td>+0.70</td>
</tr>
<tr>
<td>high</td>
<td>+0.97</td>
<td>medium</td>
<td>0.00</td>
<td>0</td>
<td>+1.47</td>
<td>-0.25</td>
</tr>
<tr>
<td>high</td>
<td>+0.97</td>
<td>low</td>
<td>-0.97</td>
<td>0</td>
<td>+1.47</td>
<td>-1.22</td>
</tr>
<tr>
<td>medium</td>
<td>0.00</td>
<td>high</td>
<td>+0.97</td>
<td>0</td>
<td>+0.50</td>
<td>+0.77</td>
</tr>
<tr>
<td>medium</td>
<td>0.00</td>
<td>medium</td>
<td>0.00</td>
<td>0</td>
<td>+0.50</td>
<td>-0.25</td>
</tr>
<tr>
<td>medium</td>
<td>0.00</td>
<td>low</td>
<td>-0.97</td>
<td>0</td>
<td>+0.50</td>
<td>-1.22</td>
</tr>
<tr>
<td>low</td>
<td>-0.97</td>
<td>high</td>
<td>+0.97</td>
<td>0</td>
<td>-0.47</td>
<td>-1.22</td>
</tr>
<tr>
<td>low</td>
<td>-0.97</td>
<td>medium</td>
<td>0.00</td>
<td>0</td>
<td>-0.47</td>
<td>-1.22</td>
</tr>
<tr>
<td>low</td>
<td>-0.97</td>
<td>low</td>
<td>-0.97</td>
<td>0</td>
<td>-0.47</td>
<td>-1.22</td>
</tr>
</tbody>
</table>
## Conditional Probability Table

| Mathematical Language | Manipulate Fractions | $Pr(X_{ij} = 0|pa(X_{ij})$ | $Pr(X_{ij} = 1|pa(X_{ij})$ | $Pr(X_{ij} = 2|pa(X_{ij})$ |
|-----------------------|----------------------|-----------------------------|-----------------------------|-----------------------------|
| high                  | high                 | 0.019                       | 0.229                       | 0.752                       |
| high                  | medium               | 0.047                       | 0.576                       | 0.377                       |
| high                  | low                  | 0.068                       | 0.828                       | 0.104                       |
| medium                | high                 | 0.083                       | 0.195                       | 0.722                       |
| medium                | medium               | 0.205                       | 0.480                       | 0.314                       |
| medium                | low                  | 0.275                       | 0.644                       | 0.081                       |
| low                   | high                 | 0.664                       | 0.299                       | 0.038                       |
| low                   | medium               | 0.664                       | 0.299                       | 0.038                       |
| low                   | low                  | 0.664                       | 0.299                       | 0.038                       |
Local Q-matrix

- In GPC models transitions can use a subset of variables.
- Q is a logical matrix with rows corresponding to state transitions, and columns to parent variables
  - True if parent is relevant for that transition
- Takes advantage of R logical subscripts
- Q=TRUE is shorthand for all variables relevant for all transitions
Open Implementation Protocols in \textit{R}

- R is a functional language, so functions (or list of functions) can be passed as arguments and stored as fields in objects.
  - CPTtools implementation allows link and combination rules to be passed in as functions
- R has an object oriented layer, so generic functions can be specialized for implementations
- Use rather loose S3 class system, which allows building new object oriented classes on top of existing RNetica implementation
Object Model

Parameterized Network
- prior weight: numeric
- GEMfit(cases: table)
- Build All Tables()
- calc log likelihood(cases: table): numeric
- calc expected tables(cases: table)
- max all table params()

Parameterized Node
- prior weight: numeric
- Q: numeric
- log alphas: numeric
- betas: numeric
- rules: function
- link: function
- link scale: numeric
- prior: function
- alphas(): numeric
- parent tvals()
- build table()
- max CPT params()
Lists and Vectors of Parameters

- R supports vectors (same type) and lists (any type)
- *Vectors* are used to indicate replication based on number of parameters (slope or intercept)
- *Lists* are used to indicate replication based on state transition (intercepts and slope and combination rules under GPC link)
Generalized EM algorithm

• E-step – Calculate expected value of sufficient statistics
  − Sufficient statistics in this case are the tables of counts of parent variable and child
  − Many BN packages (e.g., Netica) provide built-in EM algorithm supporting hyper-Dirichlet (unparameterized) model
  − Expected value of sufficient statistic is CPT output from this algorithm, weighted by row counts (Netica calls this Node experience)
  − Don’t need to run internal EM algorithm to conclusion, one step should be fine.

• M-step find parameter values that maximize sufficient statistic
  − Can do this node by node
  − Don’t need to run to convergence (generalized EM algorithm).
GEMfit

1) calcExpTables – calls internal EM algorithm (with case data) to perform E-step
2) maxAllTableParams – finds new parameters for each Pnode
3) BuildAllTables – Rebuilds the tables, and sets the weight to priorWeight
4) calcPnetLLike – Calculated the log-likelihood of data

   Algorithm ends when change in log-likelihood is less than tolerance

   All these functions are generic, so can be customized for different BN packages
Tuning parameters

- priorWeight given to Pnet CPTs in E-step
- Number of iterations taken in E-step (1 should be sufficient)
- Number of iterations taken in M-step (5 seems good)
- Convergence Tolerance
Parameter Recovery

• Depends on number of cases

• Depends on how well latent variables are identified in those cases (amount of evidence for the latent variables): test length
Package Structure: Minimizing dependence on Netica

- CPTtools – Basic calculation routines, BN implementation independent
- Peanut – OO layer for Pnet/Pnode classes
- RNetica – A specific BN implementation (Netica bound in R)
- PNetica – Peanut implementation in RNetica
Availability

- [http://pluto.coe.fsu.edu/RNetica](http://pluto.coe.fsu.edu/RNetica)
- RNetica requires Netica API license (for non-trivial examples)
- Other Bayes net package would need to support:
  - an EM learning function for hyper-Dirichlet models
  - specifying hyper-Dirichlet priors for each CPT
  - recovering the hyper-Dirichlet posteriors after running the internal EM algorithm.