Particle Swarm Optimization for Diagnosis Task: An Exploratory Study

Zhehan Jiang[1], Jihong Zhan[2], Dexin Shi[3]
The University of Alabama, USA[1],
University of Kansas, USA[2],
University of South Carolina, USA[3]

Abstract

A log-linear cognitive diagnosis model (LCDM) is estimated via a global optimization approach-particle swarm optimization (PSO), which is an efficient method of handling local maxima problem. The application of the PSO to LCDM estimation is introduced, explicated, and evaluated via a Monte Carlo simulation study in this article. The aim of this paper is to fill the gap between the field of psychometric modeling and machine learning estimation techniques.

1. Introduction

Cognitive diagnostic models (CDMs) have been developed to identify whether a student masters each attribute required for solving corresponding items. For instance, Addition, Substation, Multiplication, and Division are four common attributes defined in math ability assessment practice, where test items such as "2+4-1" measure the first two attributes and "4x2/3" measure the last two attributes. In addition to educational testing, CDMs are useful in psychological measurement. For example, literature indicates that neuro-vegetative symptoms (NS) are a general construct that contains three attributes: depression (DEP), fatigue (FAT), and sleeplessness (SLE)Using CDMs allows researchers/practitioners to investigate the attributes for a given patient. Applied works can be found in more topics, for example, Stefanutti, Anselmi, and Robusto [1] uses the CDM framework to construct leaning map.

To ensure the reliability of a measure, multiple items are always preferred. To illustrate, the Medical College Admission Test (MCAT), which is designed to measure the mastery of four attributes, biology, physical, verbal, and writing, contains more than 50 items for each attribute. Among the item data types, binary scale is the most common one that has been adopted in many surveys and measures.

Recent advances in model development have produced general diagnostic models, for instance, generalized Deterministic Input; Noisy "And" gate model (G-DINA), General Diagnostic Model (GDM) [2], and Log-linear Cognitive Diagnosis Model (LCDM) [3].

A LCDM (G-DINA or GDM) provides great flexibility such as 1) subsuming most latent variables, 2) enabling both additive and non-additive relationships between attributes and items simultaneously, and 3) syncing with other psychometric models, increasing insightfulness.

It can be proved that LCDM can be converted to core CDMs such as Deterministic Input; Noisy "And" gate (DINA), Noisy Input; Deterministic "And" gate model (NIDA), and the Reparameterized Unified Model, whereas examples of disjunctive models include the Deterministic Input; Noisy "Or" gate model. Throughout the article, the general diagnostic model is referred as a LCDM for consistency purpose.

2. The Log-linear Cognitive Diagnosis Model

As a member of latent class models, a LCDM is mathematically defined as:

$$P(\boldsymbol{Y}_p = \boldsymbol{y_p}) = \sum_{c=1}^{N_c} v_c \prod_{i=1}^{N_i} \pi_{ci}^{y_{pi}} (1-\pi_{ci})^{1-y_{pi}} \qquad (1)$$

where $\boldsymbol{y}_p = (y_{p1}, y_{p2}, \dots, y_{pI}$ is the correct/incorrect response vector of respondent p on a test comprised of N_i items, and element y_{pi} is the corresponding response on item i. v_c is the probability of membership in latent class c, and π_{ci} is the probability of correct response to item i by respondent p who belongs to class c.

Suppose there are N_α attributes. The cognitive state of a respondent is denoted by attribute vector $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_{N_\alpha})$, where each element in $\boldsymbol{\alpha}$ is a 1/0 binary variable indicating whether a respondent has mastered ath attribute α_a. There are a total number of 2^{N_α} possible attribute patterns (i.e., classes).

To illustrate, a respondent p with a pattern $\boldsymbol{\alpha} = (0, 1, 1, 0)$ has mastered the second and the third attributes, but not the first and the forth ones. Similarly, if the pattern becomes $\boldsymbol{\alpha} = (1, 1, 1, 1)$, it means the respondent has mastered all attributes.

Table 1. Formula Expression Example of a Log-linear Cognitive Diagnosis Model

Item	α_1	α_2	α_3	Complete $\lambda_{i,0} + \boldsymbol{\lambda}_i^T \boldsymbol{h}(\boldsymbol{\alpha_c}, \boldsymbol{q_i})$ Expression	Simplified Expression
1	1	0	0	$\lambda_{1,0} + \lambda_{1,1}(1) + \lambda_{1,2}(0) + \lambda_{1,3}(0) + \lambda_{1,12}(1*0) + \lambda_{1,13}(1*0) + \lambda_{1,23}(0*0) + \lambda_{1,123}(1*0*0)$	$\lambda_{1,0} + \lambda_{1,1}(1)$
2	0	1	1	$\lambda_{2,0} + \lambda_{2,1}(0) + \lambda_{2,2}(1) + \lambda_{2,3}(1) + \lambda_{2,12}(0*1) + \lambda_{2,13}(0*1) + \lambda_{2,23}(1*1) + \lambda_{2,123}(0*1*1)$	$\lambda_{2,0} + \lambda_{2,2}(1) + \lambda_{2,3}(1) + \lambda_{2,23}(1)$
3	1	1	1	$\lambda_{3,0} + \lambda_{3,1}(1) + \lambda_{3,2}(1) + \lambda_{3,3}(1) + \lambda_{3,12}(1*1) + \lambda_{3,13}(1*1) + \lambda_{3,23}(1*1) + \lambda_{3,123}(1*1*1)$	$\lambda_{3,0} + \lambda_{3,1}(1) + \lambda_{3,2}(1) + \lambda_{3,3}(1) + \lambda_{3,12}(1) + \lambda_{3,13}(1) + \lambda_{3,23}(1) + \lambda_{3,123}(1)$

To identify attributes that are required to solve each item, content experts provide a Q-matrix of size $N_i * N_\alpha$, where N_i and N_α are the numbers of items and attributes in a test respectively. The (i, a) entry of the Q-matrix q_{ia} is 1 when item i is associated with attribute a, and otherwise $q_{ia} = 0$. Given respondent p's attribute pattern is $\boldsymbol{\alpha_c}$, the conditional probability of item i can be stated as:

$$\pi_{ci} = P(y_{pi}|\boldsymbol{\alpha_c}) = \frac{exp(\lambda_{i,0} + \boldsymbol{\lambda}_i^T \boldsymbol{h}(\boldsymbol{\alpha_c}, \boldsymbol{q_i}))}{1 + exp(\lambda_{i,0} + \boldsymbol{\lambda}_i^T \boldsymbol{h}(\boldsymbol{\alpha_c}, \boldsymbol{q_i}))} \quad (2)$$

Where $\boldsymbol{q_i}$ is the set of Q-matrix entries for item i, $\lambda_{i,0}$ is the intercept parameter, where $\boldsymbol{\lambda_i}$ represents a vector of size $(2^{N_\alpha} - 1) * 1$ that contains main effect and interaction effect parameters of item i, and $\boldsymbol{h}(\boldsymbol{\alpha_c}, \boldsymbol{q_i})$ is a vector of size $(2^{N_\alpha} - 1) * 1$ with linear combinations of the $\boldsymbol{\alpha_c}$ and $\boldsymbol{q_i}$. Particularly, $\boldsymbol{\lambda}_i^T \boldsymbol{h}(\boldsymbol{\alpha_c}, \boldsymbol{q_i})$ inside the exponent function can be expressed as:

$$\boldsymbol{\lambda}_i^T \boldsymbol{h}(\boldsymbol{\alpha_c}, \boldsymbol{q_i}) = \sum_{a=1}^{N_\alpha} \lambda_{i,1,(a)} \alpha_{ca} q_{ia} + \sum_{a=1} \sum_{a'>1}^{N_\alpha} \lambda_{i,2,(a,a')} \alpha_{ca} \alpha_{ca'} q_{ia}\, q_{ia'} + \cdots, \quad (3)$$

Where $\lambda_{i,1,(a)}$ and $\lambda_{i,2,(a,a')}$ are the main effect for ath attribute α_a and a two-way interaction effect for α_a and $\alpha_{a'}$. Since elements of $\boldsymbol{\alpha_c}$ and $\boldsymbol{q_i}$ are binary, $\boldsymbol{h}(\boldsymbol{\alpha_c}, \boldsymbol{q_i})$ contains binary elements, which indicate effects needed to be estimated. For an item measuring A attributes, A-way interaction effects should be specified in $\boldsymbol{h}(\boldsymbol{\alpha_c}, \boldsymbol{q_i})$. *Table 1* shows a concrete example of a measure with 3 attributes: Item 1 that measures α_1 only has two estimates, where Item 3 measuring all three attributes has 8 estimates in total.

3. Current Estimation Methods and Problems

A variant of CDMs, LCDMs have been estimated using various algorithms, including (1) the expectation maximization (EM) [4] and (2) Markov chain Monte Carlo (MCMC) algorithms that are featured as the main estimation genres [5]. As a saturated version of CDMs, estimating LCDMs encounter more difficulties than simpler variants, such as DINA and DINO.

The EM algorithm is an iterative algorithm that involves two steps at each iteration; The E (expectation) step for computing individual's class membership and so forth the probability of membership at Equation 1, and the M (maximization) step for estimating item parameters (i.e., $\boldsymbol{\lambda}$ at Equations 2 and 3). The E and M steps are repeated until the parameter estimates converge to the maximum (log-) likelihood. The EM algorithm is reasonably fast and thus frequently used in practice. However, the increment of the number of latent classes would slow down the computation dramatically. On the other hand, the MCMC algorithms have become popular over the past decades [6] [7], and the idea has been implemented in CDMs [8]. Jiang and Carter use MCMC to estimate LCDMs [9]. Compared with the EM algorithm, at many occasions the MCMC algorithms take longer to converge because the estimation relies on random sampling that involves accept/reject decision. Unlike the EM algorithm, whose updating is directional, there are a certain number of iterations in the MCMC algorithms making no progress on the estimation process. Compared with MCMC algorithms, the EM one has more users due to its wide integration in statistical software packages.

Both genres of algorithms suffer inappropriate convergence issue without parameter constraints, meaning that the LCDM estimation is likely to encounter (1) local maxima and (2) label switching problems [10]. To be concrete, there will be multiple local maxima of the log-likelihood function that trap the algorithms. Particularly, the EM algorithm is known to converge at local maxima instead of global maxima, where only the latter provides legitimate estimates. Label switching, on the other hand, leads to

unreasonable interpretations of item parameters as well as disruption of the converging process. Rupp, Templin, and Henson [11] outlined the parameter constraint approach, for example, ensuring the positive-ness of $\boldsymbol{\lambda_{i,1}}$in Equation 3 and forcing the 2-way interaction effect $\lambda_{i,2,(a,a')}$ to be bigger than the corresponding negative main effects -$\lambda_{i,1,(a)}$ and -$\lambda_{i,1,(a')}$. Evidence suggests that the parameter constraint approach would decrease of risk of reaching local maxima and keeping label consistency [11], however, the constrained true sampling space remains unknown due to mathematical complexity. In addition, current estimation methods do not fully utilize high performance computing facility that has been rapidly developed. To reach global maxima in a derive-free mean as well as perform estimation in a parallel computing platform, a novel algorithm based upon particle swarm optimization is proposed to estimate LCDMs. In the following sections, the proposed algorithm is described thoroughly as well as its application to LCDM estimation.

4. Particle Swarm Optimization

The Particle Swarm Optimization (PSO) is a stochastic algorithm that belongs to the Swarm Intelligence methods family. Inspired by the social behavior of bird flocking and fish schooling, Eberhart and Kennedy [12] proposed PSO to find solutions to optimization problems, such as numerical integration and the travelling salesman problem [13], [14]. The term 'particle' represents a natural agent that possesses swarm behaviors (i.e., the ability of performing social interaction). Examples of swarm behaviors include (1) improving themselves to expected levels and (2) interacting with their neighborhood. In the sense of estimation, each particle stochastically explores permissible space to yield the optimal solution. The PSO is particularly useful when solutions do not exist analytically or specifically have been proven to be theoretically intractable.

The strategy of the PSO algorithm is outlined as follows: each particle l represents a candidate solution to the optimization problem in a D-dimensional space, where the particle' current solution and velocity are presented by $\boldsymbol{x}_l = (x_{l1}, x_{l2}, \dots, x_{lD})$ and $\boldsymbol{vel}_l = (vel_{l1}, vel_{l2}, \dots, vel_{lD})$. Note that in parallel computing framework, each particle can be allocated to a processing unit such that multiple particles can be executed simultaneously. The velocity is the updating step of a particle from its current solution to a future one. In addition, as the PSO algorithm stores information from its iteration history, the optimal solution of particle l (local best; $x_{best_{lo}}$) and the optimal solution across all particles (global best; $x_{best_{\mathrm{g}}}$) are used to guide velocity updates. Mathematically the iterative updating of velocity and solution can be expressed as:

$$\begin{aligned} \mathrm{vel}_{\mathrm{lj}}^{\mathrm{t}} &= \mathrm{w}^{\mathrm{t}} * \mathrm{vel}_{\mathrm{lj}}^{\mathrm{t}-1} + \mathrm{c}_1 \mathrm{r}_1^{\mathrm{t}} \left(\mathrm{x}_{\mathrm{best}_{\mathrm{lj}}}^{\mathrm{t}-1} - \mathrm{x}_{\mathrm{lj}}^{\mathrm{t}-1} \right) \\ &\quad + \mathrm{c}_2 \mathrm{r}_2^{\mathrm{t}} \left(\mathrm{x}_{\mathrm{best}_{\mathrm{gj}}}^{\mathrm{t}-1} - \mathrm{x}_{\mathrm{lj}}^{\mathrm{t}-1} \right), \quad (4) \\ \mathrm{x}_{\mathrm{lj}}^{\mathrm{t}} &= \mathrm{vel}_{\mathrm{lj}}^{\mathrm{t}} + \mathrm{x}_{\mathrm{lj}}^{\mathrm{t}-1}, \end{aligned}$$

where x_{lj}^t and vel_{lj}^t are the solution and the velocity of particle l respectively at iteration t. Parameters c_1 and c_2 are learning/acceleration factors for each term exclusively situated in the range of 2 to 4. Parameter w^t is called inertia weight ($0 \leq w \leq 1$) that can be adaptively changed along iterations. Finally, $r_1$and $r_2$are random numbers sampled from 0 to 1 independently. Tremendous variants of the PSO (hybrid PSO) have been proposed to improve the algorithm performance, for example, manipulating parameters $\boldsymbol{c}$ and w, particle mutation, and adaptively tuning velocity [15], [16]. The next section outlines the customized PSO to estimate LCDMs.

5. Hybrid PSO-EM Algorithm

The proposed algorithm is called the hybrid PSO-EM (HPSOEM) algorithm. As the name indicates, it integrates the properties of the hybrid PSO into the EM algorithm. That is, the hybrid PSO is used to replace the aforementioned M step. Pseudocode of the HPSOEM algorithm is outlined in *Figure 1*. Explanation about the steps is present in the following paragraphs.

Step 2 outlines user-defined configurations of the HPSOEM algorithm: Meeting either condition- (1) maximum iteration number or (2) minimum variance of log-likelihood of all particles' local optimum- would stop the estimation. Like any algorithm, setting the maximum iteration number is necessary in real estimation practice. The unique part of the proposed algorithm is using the minimum variance of log-likelihood of all particles' local optimum to investigate converging status. The swarm effect brings particles to the space of the optimal solution such that eventually they all end up being identical.

Providing appropriate initial values, as mentioned in Steps 3 and 4, is helpful for starting the HPSOEM algorithm.

1. Begin with Iteration t=1
2. Define algorithm parameters: (1) the number of iteration $Iter_{stop}$, (2) the number of particle N_{par}, (3) the minimum variance of log-likelihood of particles' local optimum Var_{stop}, (4) the list of model parameter constraints $List_{cons}$, (5) the penalty of violating constraints $LL_{penalty}$, (6) the maximum and minimum of the inertia weight (w_{max}, w_{min}), (7) the constraint violation tolerance $Mutate_{count}$, and (8) the particle updating parameters (c_1, c_2)
3. Initialize N_{par} D-dimensional PSOs and assign to particles' local optimum solutions $\boldsymbol{x}_{\boldsymbol{best}_l}$
4. Initialize N_{par} velocities $\boldsymbol{Vel} = \left(vel_{l1}, vel_{l2}, \ldots, vel_{lN_{par}}\right)$
5. Set all PSOs' constraint violation counts $\boldsymbol{Vio_{count}} = \left(Vio_{l1}, Vio_{l2}, \ldots, Vio_{lN_{par}}\right)$ to $\boldsymbol{0}$
6. E-step calculate the probability of membership v_c for each class
7. Calculate the log-likelihood of the initialization and assign to particles' local optimum log-likelihood $\boldsymbol{LL_{best_l}} = (LL_{best_{l1}}, LL_{best_{l2}}, \ldots, LL_{best_{lN_{par}}})$
8. Select global maximum LL_{best_g} from $\boldsymbol{LL_{best_l}}$ and its estiamtes are saved to $\boldsymbol{x}_{\boldsymbol{best_g}}$
9. While current iteration $t < Iter_{stop}$ or $Var(\boldsymbol{LL_{best\_l}}) > Var_{stop}$ do
10. Calculate inertia weight $w^t = w_{max} - \frac{w_{max} - w_{min}}{Iter_{stop}} * t$
11. For each particle l do
12. If $Vio_l > Mutate_{count}$ do
13. Current solution $\boldsymbol{x}_l^t$ is obtained by mutating from $\boldsymbol{x}_{\boldsymbol{best}_l}$ and $\boldsymbol{x}_{\boldsymbol{best_g}}$
14. Current velocity $\boldsymbol{vel}_l^t$ is re-initialized
15. Vio_l=0
16. End If
17. If $Vio_l \leq Mutate_{count}$ do
18. Current velocity $\boldsymbol{vel}_l^t$ and solution $\boldsymbol{x}_l^t$ are updated via Equation 4
19. End If
20. Update the probability of membership in latent classes $\boldsymbol{v}$
21. Calculate Log-likelihood of the current iteration LL_l^t
22. If $\boldsymbol{x}_l$ violates $List_{cons}$ do
23. $Vio_l = Vio_l + 1$ and $LL_l = LL_l + LL_{penalty}$
24. End If
25. If $LL_l^t > LL_{best_l}$do
26. $\boldsymbol{x}_{\boldsymbol{best_l}} = \boldsymbol{x}_l^t$
27. End If
28. If $LL_l^t > LL_{best_g}$do
29. $\boldsymbol{x}_{\boldsymbol{best_g}} = \boldsymbol{x}_l^t$
30. End If
31. End For
32. $t = t+1$
33. End While

Figure 1. Hybrid PSO-EM Algorithm Pseudo Code

The results obtained from the EM algorithm can be used to serve as starting values of one particle. This practice allows some particle updating to start from closer numerical space. Step 6 (and therefore Step 20) is identical to the E step in the EM algorithm; that is, condition on $\boldsymbol{\pi}$ which is obtained from the previous steps, the v_c can be updated as:

$$\hat{v}_c^t = \sum_{p=1}^{P} \frac{H(c \mid \boldsymbol{y_p})}{P}$$

$$H(c \mid \boldsymbol{y_p}) = \frac{v_c^{t-1} \prod_{i=1}^{I} \pi_{pi} (1 - \pi_{pi})^{1-y_{pi}}}{\sum_{c=1}^{N_c} v_c^{t-1} \prod_{i=1}^{I} \pi_{pi} (1 - \pi_{pi})^{1-y_{pi}}}, \qquad (5)$$

where again, subscripts t, i, c, and p represent the iteration, item, latent class, and person respectively. $H(c \mid \boldsymbol{y_p})$ is the posterior class probability for a respondent, and therefore, the marginal probability of class membership v_c is obtained by aggregating distribution on individual level.

In addition, when a solution violates the model constraint, its corresponding log- likelihood will be penalized to a certain degree. The larger the penalty is set, the less frequently the algorithm explores the solution's neighbor space. Inertia weight being set to be adaptive as Step 10 shows could "control the impact of the previous history of velocities on the current velocity and to influence the trade-off between global and local exploration abilities" of the updating particles [17]. In other words, balancing the explorations between global and local space, the adaptive strategy can effectively shorten converging time. The key element of the "hybrid" aspect is integrating the mutation idea, which is borrowed from evolutionary theory: mutation takes place when an organism needs to survive and have more offspring in a changing environment. In fact, this algorithm is named the evolutionary algorithm (EA). The essence is, if a particle has violated the model constraint for a pre-defined count consecutively, as Step 13 shows, this particle will be replaced by mutating from its local optimum and global optimum estimates, while its velocity will be reinitialized by random generation. The means of mutation can be found in EA literature [18], [19]. In this paper, mutation of a particle is created by randomly selecting a half of the solution vector from the local optimum and the other from the global optimum. Finally, steps 12-19 are the M step, while Step 20 is the E step in the sense of the EM framework.

6. Methods

The main title (on the first page) should begin 1-3/8 inches (3.49 cm) from the top edge of the page, centered, and in Times 14-point, boldface type. Capitalize the first letter of nouns, pronouns, verbs, adjectives, and adverbs; do not capitalize articles, coordinate conjunctions, or prepositions (unless the title begins with such a word). Leave two 12-point blank lines after the title.

6.1. Data Generation

A simulation study was conducted to examine the application of the HPSOEM to LCDM estimation. The simulations are based upon the Q-matrix provided in Templin and Bradshaw [13], reproduced in Table 2. There are four attributes and 28 items in total. Each item measures one or more attributes; that is, indicators can be cross-loaded in multiple latent traits simultaneously. For example, Item 1 measures the first attribute only, while Item 22 measures the second and the forth attributes. Provided the Q-matrix, LCDMs were selected to be the simulation framework.

Table 2. Q-matrix used for the Simulation Study

Item No.	Attribute1	Attribute2	Attribute3	Attribute4
1	1	0	0	0
2	0	0	1	0
3	0	1	0	0
4	1	0	0	0
5	1	0	0	1
6	0	1	0	0
7	1	0	0	0
8	0	0	1	1
9	0	0	1	0
10	0	0	1	0
11	0	0	1	0
12	1	0	0	0
13	1	0	0	1
14	1	0	0	1
15	1	0	0	1
16	1	0	0	1
17	1	0	0	0
18	0	1	0	1
19	1	1	0	0
20	0	1	0	1
21	0	1	0	1
22	0	1	0	1
23	1	0	0	0
24	1	1	0	0
25	1	1	0	0
26	0	0	1	0
27	1	0	0	0
28	1	1	0	0

6.2. Independent Variables and Dependent Variables

Responses (i.e., simulated datasets) were generated via LCDMs. Particularly, item intercepts were randomly generated from [-0.1, 1], main effects were drawn uniformly from [0.5, 5], and interaction terms were sampled from a uniform distribution, where range was [-0.5, 0.5]. The situations where item parameters violate the aforementioned constraint rules, the generation would re-start until the values produced are in permissible numeric space. Given the number of attributes is 4, and the sum of membership probabilities is 1, there are 16 classes in total and the probability of class membership was set equal (i.e., [1/16, 1/16, …, 1/16]). This class membership probability vector was set to the parameters of a multinomial distribution, which was used to sample the attributes of respondents (N_p=2000), while there are no missing responses in the simulated dataset.

The replication number was set to 500. To investigate the estimation accuracy, a parameter recovery study was conducted. In particular, the bias and the root mean squared error (RMSE) of (1) item parameters and (2) membership probability parameters were evaluated. In addition, log-likelihood values were recorded.

6.3. Software and Hardware

R environment was used to conduct the simulation study. Currently R is one of the world's most popular languages due to its cost-free property, flexible extensions, rapid package updates, and active community supports. Throughout the paper, data generation and the algorithm comparisons were executed in R. The parameters consisted in the HPSOEM were set as follow: (1) the number of particles N_{par} = 100, (2) the penalty of violating constraints $LL_{penalty}$ = log-likelihood of current iteration, (3) the particle updating parameters (c_1, c_2) = (1.5, 1.5), and (4) inertia weight (w_{max}, w_{min}) = (0.9, 0.4). These settings have been verified to work more efficiently than other combinations across various computational simulations [21], [20]. On the other hand, the CDM package. which implements the EM algorithm, was used to compare the estimation results. Note that the CDM package doesn't provide parameter constraint functions.

7. Simulation Results

As shown in the upper panel of Table 3, overall both algorithms yield appropriate item parameter estimates, while in some situations one is better than the other. For both algorithms, all absolute biases are below 0.057 and RMSEs are lower than 0.220. The HPOSEM produces slightly more accurate results, except for the main effects. It can be seen that both intercept and main effect estimates have smaller biases than interaction effect ones. These findings are consistent with Templin and Bradshaw [22]. In particular, the EM algorithm produced unsatisfactory result in the interaction effects that led the bias to above 0.05. In addition, although not demonstrated in this paper, the EM algorithm indeed produced negative values for the main effect parameters, which are prohibited in the current context.

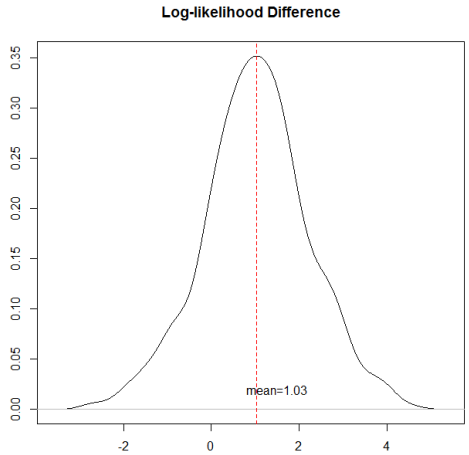

Figure 2. Distribution of Log-likelihood Difference between EM and HPSOEM

Table 3. Independent Variables of the Simulation Study

	Intercepts		Main Effects		Interaction Effects	
	EM	HPSOEM	EM	HPSOEM	EM	HPSOEM
Bias	-0.010	**0.008**	**0.005**	-0.007	0.057	**0.010**
RMSE	0.151	**0.084**	0.212	**0.155**	0.178	**0.129**

The bias for class membership probability is even more negligible (-0.002), and unsurprisingly the corresponding RMSE is 0.009. In addition to the investigation on parameters, log-likelihood difference between two algorithms is also monitored in Figure 2. Within 95% confidence interval, the difference ranges from -1.00 to 2.96. More specifically, at under 5% α-level, it can't not be concluded that the HPSOEM yields a significant larger log-likelihood value than the EM does. However, when α-level increases to 10%, the result becomes statistically significant.

8. Discussion and Conclusion

The purpose of this paper is to propose a machine-learning based algorithm for the estimation of LCDMs. In particular, the proposed estimator is a combination of the EM algorithm and the PSO techniques, which have been popular in neural networks and othersimilar fields. The performance of the proposed algorithm is evaluated through a simple simulation study of which the results indicate that it is an appropriate option to handle LCDM estimation task.

Although proved to be useful via the simulation study, it does not mean the HPSOEM is a perfect algorithm, as there isroom for the algorithm improvement. Above all, like other machine-learning estimation techniques (e.g., gradient boosting and random forest), tuning algorithm parameters is the most crucial yet difficut step when the HPSOEM is implemented. Throughout the article, the HPSOEM parameters such as inertia weights and updating parameters were chosen based upon literature recommendation. However, it is not necessarily the optimal combinations for estimating LCDMs. Beyond the scope of the present paper, a large-scale simulation may be needed to guide parameter tunning more sysmetically and efficiently. In addition, the missing data problem is not concerned here; in the future study, incorporating auxiliary information into the algorithm may help reduce estimation bias [23].

9. References

[1] L. Stefanutti, J. Heller, P. Anselmi, and E. Robusto, "Assessing the local identifiability of probabilistic knowledge structure," in Behavior Research Methods, vol. 44, no. 4, p. 1197-1211, May. 2012.

[2] M. von Davier, "A General Diagnostic Model Applied to Language Testing Data," ETS Research Report Series, vol. 2005, no. 2, p. i-35, Dec. 2005.

[3] R. A. Henson, J. L. Templin, and J. T. Willse, "Defining a Family of Cognitive Diagnosis Models Using Log-Linear Models with Latent Variables," Psychometrika, vol. 74, no. 2, p. 191, Jun. 2009.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," Journal of the Royal Statistical Society. Series B (Methodological), vol. 39, no. 1, pp. 1–38, 1977.

[5] J. L. Templin and R. A. Henson, "Measurement of Psychological Disorders using Cognitive Diagnosis Models.," Psychological Methods, vol. 11, no. 3, pp. 287–305, 2006.

[6] Z. Jiang, and W. Skorupski, "A Bayesian approach to estimating variance components within a multivariate generalizability theory framework," in Behavior Research Methods, p. 1-22, Dec. 2017.

[7] Z. Jiang, and J. Templin, "Constructing Gibbs Samplers for Bayesian Logistic Item Response Models," in Multivariate Behavioral Research, vol. 53, no. 1, p. 132, Jan. 2018.

[8] L. V. DiBello and W. Stout, "Student Profile Scoring for Formative Assessment," in New Developments in Psychometrics, Springer, Tokyo, 2003, pp. 81–92.

[9] Z. Jiang, and R. Carter, "Using Hamiltonian Monte Carlo to estimate the log-linear cognitive diagnosis model via Stan," in Behavior Research Methods, p. 1-22. Jun. 2018.

[10] H. Lao, "Estimation of Diagnostic Classification Models without Constraints: Issues with Class Label Switching," Thesis, University of Kansas, 2016.

[11] A. Rupp, J. Templin, and R. Henson, "Diagnostic Measurement: Theory, Methods, and Applications," Guilford Press, New York, NY, 2010.

[12] J. Kennedy and R. C. Eberhart, "A Discrete Binary Version of the Particle Swarm Algorithm," in Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on, 1997, vol. 5, pp. 4104–4108.

[13] M. Dorigo and L. M. Gambardella, "Ant Colonies for the Travelling Salesman Problem," Biosystems, vol. 43, no. 2, pp. 73–81, Jul. 1997.

[14] L. Djerou, N. Khelil, and M. Batouche, "Numerical Integration Method Based on Particle Swarm Optimization," in Advances in Swarm Intelligence, 2011, pp. 221–226.

[15] N. Ben Guedria, "Improved Accelerated PSO Algorithm for Mechanical Engineering Optimization Problems," Applied Soft Computing, vol. 40, no. Supplement C, pp. 455–467, Mar. 2016.

[16] M. Maitra and A. Chatterjee, "A Hybrid Cooperative–Comprehensive Learning Based PSO Algorithm For Image Segmentation Using Multilevel Thresholding," Expert Systems with Applications, vol. 34, no. 2, pp. 1341–1350, Feb. 2008.

[17] S. Kim and L. Li, "A Novel Global Search Algorithm for Nonlinear Mixed-effects Models using Particle Swarm Optimization," J Pharmacokinet Pharmacodyn, vol. 38, no. 4, pp. 471–495, Aug. 2011.

[18] Q. Zhang, J. Sun, and E. Tsang, "An Evolutionary Algorithm With Guided Mutation for the Maximum Clique Problem," IEEE Transactions on Evolutionary Computation, vol. 9, no. 2, pp. 192–200, Apr. 2005.

[19] R. Shukla, B. Hazela, S. Shukla, R. Prakash, and K. K. Mishra, "Variant of Differential Evolution Algorithm," in Advances in Computer and Computational Sciences, Springer, Singapore, 2017, pp. 601–608.

[20] A. R. Jordehi and J. Jasni, "Parameter Selection in Particle Swarm Optimisation: a Survey," Journal of Experimental and Theoretical Artificial Intelligence, vol. 25, no. 4, pp. 527–542, Dec. 2013.

[21] S. Kessentini and D. Barchiesi, "Particle Swarm Optimization with Adaptive Inertia Weight," International Journal of Machine Learning and Computing, vol. 5, no. 5, pp. 368–373, Oct. 2015.

[22] J. Templin and L. Bradshaw, "Hierarchical Diagnostic Classification Models: A Family of Models for Estimating and Testing Attribute Hierarchies," Psychometrika, vol. 79, no. 2, pp. 317–339, Apr. 2014.

[23] Z. Jiang, K. Walker, D, Shi, and J. Cao, "Improving generalizability coefficient estimate accuracy: A way to incorporate auxiliary information," in Methodological Innovations, p. 1-14, Aug. 2018.

10. Acknowledgements

This work was supported by Research and Economic Development at the University of Alabama [grant number RG14790].

