Upgrade Manual

for Latent GOLD 5.1[1]

Jeroen K. Vermunt and Jay Magidson

Version: January 9, 2016

Statistical Innovations Inc.

www.statisticalinnovations.com

[1]This document should be cited as "J.K. Vermunt and J. Magidson (2016). Upgrade Manual for Latent GOLD 5.1 Belmont Massachusetts: Statistical Innovations Inc."

Contents

1 Introduction

In January of 2016, we formally released version 5.1 of Latent GOLD®. The major new feature in this release is that it is available as a 64-bit program. This removes the limitation of a maximum of 2GB RAM usage and makes it possible to run even more complex models with even larger data sets.

Beginning with this release, we also simplified the structure of the program so that it now consists of a Basic program and two add-ons (Choice and Adv/Syntax), as well as the possibility to obtain either an annual or a perpetual license. In addition, annual Student licenses for fulltime Masters and PhD students, and Classroom licenses for Academic teachers are now available.

This upgrade manual describes the new features in the Basic and Adv/Syntax modules of Latent GOLD® 5.1 which were implemented in the transition from version 5.0 to version 5.1. Although no additional features were implemented in the Choice add-on module, the new Adv/Syntax features can also be used in Choice models. For complete documentation of Latent GOLD®, please also refer to our user's guide, syntax guide, technical guide, and the Latent GOLD® 5.0 upgrade manual.

Below we will first provide more details about the new program structure. Then we describe the new features that apply to all Latent GOLD® models, followed by the new Basic and Syntax features.

2 New program structure

Beginning with version 5.1, we simplified the Latent GOLD® program structure. This involves three changes:

1. Regardless of the program licensed, all versions of Latent GOLD® 5.1 contain the basic program, consisting of the Cluster, DFactor, Regression, and Step3 modules.

2. Latent GOLD® Choice is no longer a stand-alone program. Instead, Choice is now an optional add-on to Latent GOLD® 5.1 Basic. Thus, Choice users will not only have access to the Choice module, but also to the four Basic modules.

3. The Advanced and Syntax add-ons from earlier releases of Latent GOLD® are merged into a single optional Adv/Syntax add-on for Latent GOLD®

5.1 Basic. This add-on integrates Advanced options (e.o. multilevel modeling, continuous factors, and complex sampling) with the Basic modules, as well as the Markov module, and allows use of the Syntax module.

These changes result in 4 program options:

1. Latent GOLD® 5.1 Basic
2. Latent GOLD® 5.1 Basic + Adv/Syntax
3. Latent GOLD® 5.1 Basic + Choice
4. Latent GOLD® 5.1 Basic + Choice + Adv/Syntax

Each of these Latent GOLD® 5.1 programs can be licensed on either an annual or a perpetual basis.

In addition to accommodating Academic and Non-Academic (Commercial and Government) users, we now also offer Student licenses to accommodate fulltime Masters and PhD students, and Classroom licenses to accommodate Academic teachers who wish to provide Latent GOLD® access to members of their class.

3 All models

Changes that affect all Latent GOLD® models concern the 64-bit processing, the new “Paired Comparisons” output, and the use of the command line version of the program.

3.1 64-bit version

A limitation of the 32-bit version of Latent GOLD® was that it could use no more than 2GB RAM, whereas current computers may have 16GB RAM or even more at their disposal. The 64-bit version of Latent GOLD® removes this limitation, implying that you will be able to use all available RAM. Though in many applications 2GB RAM is sufficient, many users are now analyzing larger data sets and running bigger and more complex models, which may require more than 2GB RAM. Moreover, the multiple processing procedure implemented since version 5.0 for speeding up estimation also requires more memory since each of the threads will have its own copy of the internal arrays used during the computations. For example, suppose your computer has an 8-core CPU and you want to run a model requiring 300MB RAM per thread. Since this requires 2.4GB RAM, this will not run with the 32-bit version of the program if you wish to use all 8 cores. This problem will no longer occur with the 64-bit version so long as your computer has sufficient RAM.

3.2 Wald tests for paired comparisons between classes

The Latent GOLD® Parameters output reports overall Wald tests for the differences in parameters between classes –e.g., for Cluster-Indicator associations in Cluster models, Class-specific predictor effects in Regression or Choice models, and Covariate-Class effects in any type of Latent GOLD® model. However, often we may also wish to check which *specific* classes differ from each other. For this purpose, we implemented a new type of Wald test in which the parameters of interest are compared across pairs of latent classes. These tests are especially useful when the null-hypothesis of the corresponding overall Wald test in the Parameters output is rejected and you would like to know which classes differ from one another.

This new output allows, for example, testing in a Cluster model which Clusters are significantly different in terms of the indicators, in a Regression model which Classes have significantly different predictor effects, or in a Step3 model for which pairs of Classes the effects of the Covariates are significantly different. Note that when the term concerned contains nominal variables, the number of parameters involved in the pairwise comparisons will be larger than one, which is reflected in the number of degrees of freedom of the test.

This output can be seen by expanding the Parameters output (by clicking on the + sign next to Parameters) and selecting Paired Comparisons. For parameters whose values are allowed to differ across latent classes, you obtain a separate Wald test for each pair of latent classes. A significant p-value associated with this Wald statistic means that the parameters differ between the classes concerned in a statistically significant way.[2]

[2]Note that typically one may wish to account for multiple testing, for example, by applying a Bonferroni-correction which involves dividing the type I error rate by the number of tests (by the number of pairs of classes).

3.3 Running the program in batch mode from command line

A few changes have been implemented compared to the Latent GOLD® 4.5 and 5.0 versions concerning the way models are run in batch mode. The full command line specification in Latent GOLD® 5.1 is:

```
lg51 <modelfile> /b /h /o <outputfile> /l
```

The most important change is that one should use the switch /b to run a model in batch model. Excluding this switch will simply open the model file in the Latent GOLD® GUI (Graphical User Interface) rather than starting a batch run. The /h switch, which is not new, indicates the output should be in html instead of text format.

Two new options are available via the /o and /l (lowercase L) switches. The /o switch followed by a file name can be used to set the name of the output file (by default it is derived from the model file name). The /l switch indicates that the log (or iteration detail) information should be shown on the screen, which may be useful if one wishes to see how the estimation is progressing.

4 Basic models

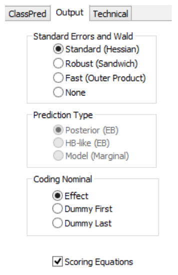
In this section, we describe the two options implemented in Latent GOLD® Basic. The first option is available for cluster models and the second for cluster and DFactor models.

4.1 Cluster: 1-Step Scoring

A new feature in version 5.1 is the ability to obtain scoring equations for latent class (LC) cluster models. The scoring equations in tabular form appear with the other output *immediately* after estimating a LC cluster model. We refer to this as 1-Step Scoring to distinguish it from the more general scoring procedure that is implemented in the Step3 module. While the same scoring equation can also be obtained using the Step3-scoring analysis, the Step3 analysis requires an additional step. Thus, scoring equations for many

LC cluster models can now be obtained much more quickly than before (For an introduction to scoring equations and how to use them to score new cases see Step3 tutorial #2.).

Except for multilevel LC cluster models or LC cluster models containing continuous factors (CFactors), 1-Step Scoring is available in the Basic version of Latent GOLD for *any* LC cluster model. To obtain the scoring equations, prior to estimating the LC cluster model simply check the Scoring Equations box on the Output tab as shown in Figure 1:



Standard Errors and Wald: [✓ Standard (Hessian)] Robust (Sandwich) Fast (Outer Product) None

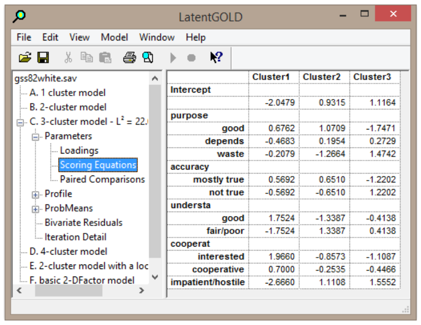
Prediction Type: [✓ Posterior (EB)] HB-like (EB) Model (Marginal)

Coding Nominal: [✓ Effect] Dummy First Dummy Last

[✓ Scoring Equations]

Figure 1: Output tab showing selection of Scoring Equations

Following model estimation, the Scoring Equations appear as a subsection of the Parameters Output as shown in Figure 2.



	Cluster1	Cluster2	Cluster3
Intercept			
	-2.0479	0.9315	1.1164
purpose			
good	0.6762	1.0709	-1.7471
depends	-0.4683	0.1954	0.2729
waste	-0.2079	-1.2664	1.4742
accuracy			
mostly true	0.5692	0.6510	-1.2202
not true	-0.5692	-0.6510	1.2202
understa			
good	1.7524	-1.3387	-0.4138
fair/poor	-1.7524	1.3387	0.4138
cooperat			
interested	1.9660	-0.8573	-1.1087
cooperative	0.7000	-0.2535	-0.4466
impatient/hostile	-2.6660	1.1108	1.5552

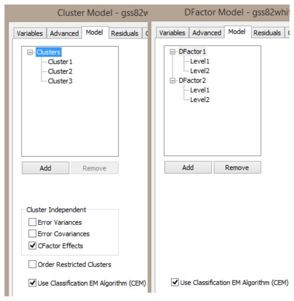
Figure 2: Scoring Equations output in tabular form

Note: Scoring equations are also attainable for LC cluster models estimated using the Classification EM algorithm (CEM).

4.2 Cluster and DFactor: Classification EM algorithm to perform extended K-Means clustering

LC clustering can be viewed as a probabilistic extension of the popular K-Means clustering algorithm for continuous indicators (Magidson and Vermunt, 2002). While K-Means clustering uses Euclidean distance to assign cases to the nearest cluster, LC clustering uses a probabilistic definition of distance which applies not only to continuous but also categorical indicators. A new feature available in Latent GOLD 5.1® Advanced/Syntax is the ability to obtain the K-Means clustering plus various extended K-Means type solutions, by using the *classification* EM algorithm (CEM) which maximizes the classification log-likelihood (CLL) rather than the standard log-likelihood (LL). For further details, see section 5.1 below.

To use the CEM algorithm simply check Use the Classification EM algorithm (CEM) on the Model tab as shown in Figure 3.



Cluster Independent
- [ ] Error Variances
- [ ] Error Covariances
- [x] CFactor Effects

- [ ] Order Restricted Clusters
- [x] Use Classification EM Algorithm (CEM)

- [x] Use Classification EM Algorithm (CEM)

Figure 3: Model tab showing selection of CEM algorithm for LC Cluster (left) and DFactor (right) models

When the CEM algorithm is used and Classification-Posterior output is requested from the Output tab, the posterior probabilities displayed in the Classification output will show that each case is assigned to one of the classes (or one level of each DFactor) with probability 1, indicative of the hard partitioning provided by K-Means and extended K-Means models.

5 Syntax models

In this section, we describe the new options implemented in Latent GOLD® Syntax. Some of these options are in a somewhat experimental stage. Despite this we make them available because we think they may be useful for some users.

5.1 Algorithm: Classification EM algorithm

We implemented a new type of estimation procedure, that is, the classification Expectation-Maximization (EM) algorithm. This is a variant of EM which maximizes the classification log-likelihood (Celeux and Govaert, 1992). The procedure is similar to K-means clustering since it yields a hard-partitioning of the sample (Vermunt, 2011), and in fact can provide a K-means solution as a special case when all variables are continuous. It can also provide many generalized solutions involving continuous and/or categorical scale types. To use this feature, include the keyword “cem” as one of the algorithm options.

The CEM-based hard-partitioning is applied to the latent variables at the highest level of the specified model. Usually these are the Classes at the case level. However, in multilevel latent class models, these are the GClasses at the group level. The hard-partitioning is never applied to the dynamic states in latent Markov models.

5.2 Startvalues

Three new features were added to influence the Latent GOLD® starting values procedure:

- nriterations=%d: inclusion of this option causes Newton iterations (in addition to EM) to be used in the startvalues procedure

- PCA: this option yields starting values for factor loading based on a Principal Components Analysis (PCA) of the full sample. The starting values for these parameters will be the PCA loading plus some random variation.

- anneal: this option implements the annealing EM algorithm for the start sets. In the Latent GOLD® implementation, it involves making the posterior class memberships more fuzzy during the initial 100 EM iterations. The fuzzyness is largest during the first iterations and reduced every 5 iterations. This option may be useful when local maxima result from posteriors being too extreme during these initial iterations.

5.3 Bayes

There are three new Bayes options available in Latent GOLD$^{®}$ Syntax models:

- perclass: This indicates that the numbers specified with the other Bayes options represent the amount of pseudo-data added to each class (instead of the *total* amount of pseudo-data)
- percategory: This indicates that the number specified with the *categorical* option represents the average amount of pseudo-data added per category of the categorical variables. For dichotomous variables it is equivalent to using categorical=2, for trichotomous variables to categorical=3, etc.
- uniform: This indicates that the number specified with the *categorical* option is divided equally across categories, rather than based on the marginal distribution of the variable concerned. So, in fact, it means that a Dirichlet prior is used with a constant parameter α.

Using all three options together yields a prior with a constant $\alpha = 1$ for each class-category combination.

5.4 Step3: norescale option

By default the posteriors specified when defining a step3 model are rescaled to sum to 1 (Typically, this is not necessary since the sum of the posteriors will equal one when the posteriors for all classes are used). In Syntax models, rescaling can be suppressed by using the keyword “norescale” in the step3 option. This may be useful if one specifies a step3 model using only a subset of the original latent classes.

5.5 GUI-like ordering of classes

In Latent GOLD$^{®}$ GUI models, the label switching problem is resolved by reordering the classes after the estimation algorithm converges. The latent classes will then appear in the output sorted from large to small (taking into account constraints). With the new Syntax output option “reorder”, one achieves the same result. The reordering is done for all discrete latent variables in the model, with the same treatment of ordinal latent variables as in DFactor models (reversing the order of the levels if necessary). The reorder option should always work for models without restrictions on class-specific parameters. In models with such restrictions, the program will still try to determine whether reordering is possible, but will do this in a slightly “conservative” manner to guarantee that the applied reordering does not change the log-likelihood value.

5.6 Two-level and longitudinal BVRs for other scale types

In Latent GOLD® 5.0, we implemented two-level and longitudinal Bivariate Residuals (BVRs) for categorical dependent variables. New in Latent GOLD® 5.1 is that these are provided for all scale types and that a linear-by-linear association is assumed for ordinal variables, as is also done in the standard BVRs. The two-level BVRs are provided in multilevel latent class models and the longitudinal BVRs in Markov models, both in Syntax and in GUI models. For technical details on the categorical variants, see Nagelkerke, Oberski, and Vermunt (2015). The new variants for other scale types are based on unexplained variances and unexplained correlations.

5.7 Asymptotic power computation for likelihood-ratio tests

In Latent GOLD® 4.5, we implemented Monte-Carlo-based power computations for the likelihood-ratio test with nested models. In Latent GOLD® 5.0, we implemented asymptotic power computation for Wald tests (Gudicha, Tekle, and Vermunt, in press). New in Latent GOLD® 5.1 is the computation of asymptotic power of the likelihood-ratio test using the procedure described in Gudicha, Schmittmann, and Vermunt (in press). This involves defining and running H1 and H0 models using either an exemplary data set or a large simulated data set under the H1 model. The keywords to be used in the output section of the H1 model are: LLdiff=‘H0model’ and LLdiffPower=n. LLdiff=‘H0model’ refers to the reference model H0 which contains the restriction that should be tested. LLdiffPower=n indicates the sample size to achieve a specified power (if n¡1) or the power achieved by a specified sample size (if n¿1).

5.8 Loadings for Cluster- and DFactor-like models with Syntax

In Cluster and DFactor models, Latent GOLD® provides factor analysis like loadings as an addition to the Parameters output. These are obtained using a linear approximation of the class-specific response probabilities (Vermunt and Magidson, 2005). This output can now also be obtained in Latent GOLD® Syntax models by using the new output option "loadings". Since the output is computed in the same manner as in Cluster and DFactor models, it will be correct only in models with a single nominal latent variable or with one or more ordinal latent variables at the case level. In the future, we may extend this output to other types of Syntax models, as well as to situations in which one would like to obtain this information for different subgroups.

5.9 The LTB method for distal outcomes

Lanza, Tran, and Bray (2013) proposed a method for investigating the relationship between class membership and distal outcomes, which is especially useful with continuous distal outcomes (referred to as the LTB method). It involves using the distal outcome as a (numeric) covariate in the latent class model of interest, and subsequently computing output similar to what is provided in the Profile output in Cluster models.

In Syntax models, this output is obtained using the output option "profile=LTB". It yields the class-specific means and variances for the distal outcome(s) included in the model as (numeric) covariate(s). However, standard errors and tests provided are somewhat ad hoc (see, Bakk, Oberski, and Vermunt, in press). Bakk, Oberski, and Vermunt (in press) show that correct standard errors and tests can be obtained by combining the LTB approach with bootstrap standard errors, which is achieved with the output option standarderrors=npbootstrap.

The LTB procedure can also be combined with a three-step latent class analysis. This involves using the distal outcomes as covariates in the step3 model (Bakk, Oberski, and Vermunt, in press).

5.10 Adjusted chi-squared statistics with missing data

In models for categorical dependent variables, Latent GOLD® reports goodness-of-fit chi-squared statistics. However, with missing values on the dependent variables, these statistics provide a simultaneous test to both the model of interest and the assumption that the missing data is MCAR (missing completely at random) (Vermunt, 1997). In Latent GOLD® 5.1, we implemented a simple new procedure to obtain goodness-of-fit tests under the weaker MAR (missing at random) assumption which underlies maximum likelihood estimation of the model parameters.

The procedure begins by estimating the saturated model with the records with missing data included and computing the likelihood-ratio, Pearson, and Cressie-Read chi-squared statistics for this model. Since the saturated model itself fits perfectly, the resulting chi-squared statistics test only the MCAR assumption. By subtracting the resulting chi-squared values from those of the Latent GOLD® model of interest and adjusting the number of degrees of freedom in the appropriate manner, we then obtain the corresponding chi-squared tests for the model of interest under MAR.

Chi-squared statistics adjusted for missing data are obtained with the output option "MARchi2". It should be noted that this option should not be used in tables with more than 100,000 cells, because otherwise estimation of the saturated model may take very long. Moreover, chi-squared statistics for such tables will not be very meaningful.

These chi-squared adjustments can also be used when bootstrapping the distribution of the chi-squared statistics.

5.11 Multiple imputation

Latent GOLD® 5.0 implements multiple imputation using procedures other than EM and the default nonparametric bootstrap. When using simple latent class models for categorical dependent variables, in version 5.1 one can also obtain multiple imputations using a divisive Latent Class procedure, Bayesian estimation with Gibbs sampling, and Bayesian estimation using a Dirichlet process prior (Van der Palm, Vermunt and Van der Ark, in press; Vidotto, Kaptein, and Vermunt, in press). These experimental options are invoked with the keywords "divisive", "gibbs", and "dpp" respectively in the outfile option "imputation". For the divisive procedure one should use a 2-class model, for the gibbs procedure a LC model with the number of classes one wishes, and for the dpp procedure a model with a large number of classes.

5.12 Complex sampling standard errors: single PSU in some strata

Although in theory this should not occur, when using the complex sampling standard error computation option, it may happen that some strata consist of a single PSU. We implemented three ways of dealing with this problem:

- certain: in the linearization estimator the single PSU has a contribution of 0 to covariance of the gradients; in the jackknife and the bootstrap the single PSU is always included.
- scaled: is the same as certain but the variances are inflated by the fraction of certain strata.
- centered: in the linearization estimator gradients of the single PSU are compared with the overall gradients (which equals 0); in the jackknife the single PSU is treated in the same way as the other observations; in the bootstrap it is included with probability 0.5.

These are keywords that can be used with the specification of the stratumid, where certain(ty) is the default.

5.13 Linear and log-linear models for dichotomous and ordinal dependent variables

Latent GOLD® 4.5 and 5.0 implement various types of models for ordinal dependent variables. These include the adjacent category, cumulative, and sequential logit models, as well as the probit and (complementary) log-log model. Two new types of models (link functions) are implemented in Latent GOLD® 5.1 Syntax (in experimental stage). These are log-linear and linear models for (cumulative) probabilities. The corresponding keywords to be used in the definition of the dependent variable concerned are "log" and "lin", respectively.

Note that the identity link (the linear model) does not guarantee that probabilities stay in the 0-1 range and the log link does not guarantee that probabilities are smaller than 1. Therefore, special care should be taken when specifying (user-defined) starting values. Moreover, there is no guarantee that the algorithms implemented in Latent GOLD® will converge. On the other hand, sometimes it is convenient to define constraints on probabilities in linear or log-linear form, which is why we implemented these models.

5.14 Start values and restriction for sets of parameters

In Latent GOLD® 5.1, we expanded the way in which users can specify starting values and place restrictions on parameters as part of the Equations. More specifically, starting values and constraints can now also be specified for multiple sets of parameters simultaneously. A few examples are:

- a b c ˜= {<startvalues>};
- a - c d ˜= 'name of file with starting values';
- a b c = {<fixed values>};
- a b c = 2; instead of a = 2; b = 2; c = 2;
- a b c = d; instead of a = d; b = d; c = d;
- a b c = +; a, b, and c should be non-negative.

Thus, at the left-hand side one can specify a *list* of parameters. The right-hand side remains as usual. If the number of specified values is fewer than the total number of parameters implied by the list, Latent GOLD® will return to the first value and continue reading values until the complete sequence of values is satisfied.

5.15 Providing variances of "fixed" parameters

Bakk, Oberski, and Vermunt (2014) discussed the adjustment of standard errors in the third-step model of a three-step latent class analysis by taking into account that this model uses the parameter estimates from the step-1 model, which themselves have a particular variance-covariance matrix. In fact, they use a rather general method for standard error correction in stepwise estimation. Although not straightforward to use, this procedure is also available in Latent GOLD® 5.1.

The procedure is invoked by defining fixed values for a single set or a list of parameters, and specifying the name of the file from which the fixed values can be read (see previous subsection). However, the specified file may contain more information than just the fixed values; that is, it may contain the following five items:

- Values of the fixed parameters. This is required.
- Number of original parameters in the step-1 model and the Jacobian matrix, which is the matrix of first derivatives of fixed parameters towards the step-1 parameters. This is optional. It can be used when the fixed parameters are functions of the step-1 parameters.
- Minus inverse Hessian matrix of the step-1 parameters. This is required.
- Covariance matrix of the step-1 parameters. This is optional, It is used when another kind of SE estimator is used in step-1 than the one based on minus inverse Hessian.
- Gradient contributions of all observations for the step-1 parameters. This is optional. It is used for the second-order correction described by Bakk, Oberski, and Vermunt (2014). When using this option, one should make use a caseid in both runs to make sure that the sorting of the records coincide between the two estimation steps.

Each of these five items is preceded by a keyword as follows – Step1Parameters, Step1Jacobian, Step1Hessian, Step1Variances, and Step1Gradients, respectively – and followed by the numbers. To see how the file is structured one can use the output option writeclassificationerrors= 'filename'. This yields a file with classification error logits as Step1Parameters and each of the other 4 items discussed above.

6 References

Bakk, Z., Oberski, D.L., and Vermunt, J.K. (2014). Relating latent class assignments to external variables: standard errors for corrected inference. *Political Analysis*, 22, 520-540.

Bakk, Z., Oberski, D.L., and Vermunt, J.K. (in press). Relating latent class membership to continuous distal outcomes: improving the LTB approach and a modified three-step implementation. *Structural Equation Modeling.*

Celeux, G., and Govaert, G. (1992). A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, 14, 315–332.

Gudicha, G.W., Tekle, F.B., and Vermunt, J.K. (in press). Power and sample size computation for Wald tests in latent class models. *Journal of Classification.*

Gudicha, D.W., Schmittmann, V.D., and Vermunt, J.K. (in press). Power computation for likelihood ratio tests for the transition parameters in latent Markov models. *Structural Equation Modeling.*

Lanza, T. S., Tan, X., and Bray, C. B. (2013). Latent class analysis with distal outcomes: A flexible model-based approach. *Structural Equation Modeling*, 20, 1-26.

Magidson, J., and Vermunt, J.K. (2002). Latent class models for clustering: a comparison with K-means. *Canadian Journal of Marketing Research*, 20, 36-43.

Nagelkerke, E., Oberski, D.L., and Vermunt, J.K. (in press). Goodness-of-fit of multilevel latent class models for categorical data. *Sociological Methodology.*

Van der Palm, D.W., Van der Ark, L.A., and Vermunt, J.K. (in press). Divisive latent class modeling as a density estimation method for categorical data. *Journal of Classification.*

Vermunt, J.K. (1997). *Log-linear models for event histories.* Thousand Oakes: Sage.

Vermunt, J.K.. (2011). K-means may perform as well as mixture model clustering but may also be much worse: Comment on Steinley and Brusco (2011). *Psychological Methods*, 16, 82-88.

Vermunt, J.K., and Magidson, J. (2005). Factor Analysis with categorical indicators: A comparison between traditional and latent class approaches. In A. Van der Ark, M.A. Croon, and K. Sijtsma (eds.), *New Developments in Categorical Data Analysis for the Social and Behavioral Sciences*, 41-62. Mahwah: Erlbaum.

Vidotto, D., Kaptein, M.C., and Vermunt, J.K. (2015). Multiple imputation of missing categorical data using latent class models: State of art. *Psychological Test and Assessment Modeling*, 57, 542-576.