Recommended Practices in Latent Class Analysis using the Open-Source R-Package tidySEM

Caspar J. van Lissa[1], Mauricio Garnier-Villarreal[2], & Daniel Anadria[3]

[1] Tilburg University, Methodology & Statistics

[2] Vrije Universiteit Amsterdam, Sociology

[3] Utrecht University, Methodology & Statistics

Author Note

The authors made the following contributions. Caspar J. van Lissa: Conceptualization, Software, Supervision, Writing – original draft, Writing – review & editing; Mauricio Garnier-Villarreal: Writing – review & editing; Daniel Anadria: Writing – original draft, Writing – review & editing.

Correspondence concerning this article should be addressed to Caspar J. van Lissa, Professor Cobbenhagenlaan 125, 5037 DB Tilburg, The Netherlands. E-mail: c.j.vanlissa@tilburguniversity.edu

Abstract

Latent class analysis (LCA) refers to techniques for identifying groups in data based on a parametric model. Examples include mixture models, LCA with ordinal indicators, and latent class growth analysis. Despite its popularity, there is limited guidance with respect to decisions that must be made when conducting and reporting LCA. Moreover, there is a lack of user-friendly open-source implementations. Based on contemporary academic discourse, this paper introduces recommendations for LCA which are summarized in the SMART-LCA checklist: Standards for More Accuracy in Reporting of different Types of Latent Class Analysis. The free open-source R-package package `tidySEM` implements the practices recommended here. It is easy for beginners to adopt thanks to user-friendly wrapper functions, and yet remains relevant for expert users as its models are integrated within the `OpenMx` structural equation modeling framework and remain fully customizable. The Appendices and `tidySEM` package vignettes include tutorial examples of common applications of LCA.

Keywords: latent class analysis, mixture models, recommended practices, free open-source software

Word count: 6110

Recommended Practices in Latent Class Analysis using the Open-Source R-Package tidySEM

Latent class analysis (LCA) is an umbrella term that refers to a number of structural equation modeling (SEM) techniques for estimating unobserved groups based on a parametric model of observed indicators (Vermunt et al., 2004). Its popularity has skyrocketed in recent years; it has nearly half as many search results in Google Scholar as factor analysis, a foundational SEM technique. Despite the popularity of LCA, two challenges impede broader adoption and correct application. The first is a lack of standards for conducting and reporting LCA. This introduces a risk of misapplications and complicates manuscript review and quality assessment of published studies. This paper addresses that issue by introducing updated estimation and reporting guidelines, summarized in the SMART-LCA Checklist: Standards for More Accuracy in Reporting of different Types of Latent Class Analysis. A second challenge is that most existing software is commercial and closed-source (e.g., Mplus, Muthén & Muthén, 1998; Latent GOLD, Vermunt & Magidson, 2000). Access to commercial software is restricted to license holders, and its proprietary source code cannot be audited, debugged, or enhanced by third parties. Free open-source software (FOSS), by contrast, removes financial barriers and promotes collaboration, inclusivity, and transparency (Lamprecht et al., 2020). Notable FOSS for LCA includes `mclust` (Scrucca, Fop, Murphy, & Raftery, 2016) and `OpenMx` (Neale et al., 2016), but there are still limitations pertaining to model complexity, user-friendliness, and documentation. To address these limitations, this paper introduces new LCA functionality in the `tidySEM` R-package (C. J. Van Lissa, 2022b). To ensure a low threshold for novice users, common LCA techniques are accessible through user-friendly functions, and custom models can be specified in `lavaan` syntax (Rosseel, 2012). For expert users, the package has a high ceiling, as the models are estimated in - and compatible with - `OpenMx`, a general purpose extensible SEM framework. The `tidySEM` package was developed with open science in mind, following the FAIR (Findable, Accessible, Interoperable, and Reusable) software principles (Lamprecht et al., 2020), and OpenSSF Best Practices. It implements LCA, paying particular attention

to the practices discussed in this paper.

LCA has become popular under different names across scientific fields. Some authors use the term LCA to refer specifically to *LCA with ordinal indicators*, which has introduced some ambiguity (Collins & Lanza, 2009). LCA with continuous indicators is also known as a *finite Gaussian mixture model*. This name reflects the assumption that the observed data is a mixture of distributions from multiple underlying subpopulations. For example, men's and women's shoesizes are normally distributed around different means. In a population sample, the distribution of shoe size will be a bimodal mixture of those two distributions. A two-class LCA could be used to separate these two mixed distributions and recover participants' probability of belonging to each sex based on their shoe size. A special case of the mixture model is *latent profile analysis (LPA)*, which assumes conditional independence of the indicators. LCA can also use latent variables as indicators. For example, *growth mixture models* (GMM) and *latent class growth analyses* (LCGA) use latent variables to account for repeated measures (Jung & Wickrama, 2008). The indicators of class membership are the intercepts and (co)variances of latent growth variables that describe individual trajectories on the observed variables. LCGA differs from GMM in that it assumes within-class homogeneity of trajectories, restricting the (co)variances to zero. Another longitudinal extension of LCA is the *Hidden Markov Model* or *Latent Transition Analysis*, which estimates an LCA at each time point and models the probabilities of transitioning from one class to another (Vermunt, 2010b). Models with multiple categorical latent variables are beyond the scope of this paper, however.

LCA is similar to CFA; both measure a latent variable while accounting for measurement error (Molenaar & Eye, 1994). The key distinction is that LCA assumes a categorical instead of continuous latent variable. As CFA groups *variables* into latent constructs, it has been called a "variable-centered" technique; LCA, by contrast, groups *observations* into classes and is thus referred to as a "person-centered" technique (Masyn, 2013; Nylund-Gibson & Choi, 2018). When the focus is on class-specific model parameters,

LCA bears similarities to multi-group SEM. An important distinction is that a multi-group model treats each group's data as independent samples. In LCA, by contrast, observations contribute to parameters in all classes, but the relative contribution of each case to each class is determined by its *posterior probability* of belonging to that class. When the focus is on individuals' class membership, LCA can be conceptualized as parametric *model-based clustering* (Hennig, Meila, Murtagh, & Rocci, 2015; Scrucca et al., 2016). Compared to non-parametric clustering, LCA can be relatively parsimonious and thus suitable for smaller samples. Moreover, LCA offers fit measures, interpretable parameters, and classification diagnostics. Finally, in machine learning, LCA is considered to be one of many unsupervised classification methods (Figueiredo & Jain, 2002).

To illustrate the different uses of LCA, we provide tutorial examples in the Online Supplementary Materials. The package vignettes contain up-to-date versions of these tutorials, see `vignette(package = "tidySEM")`. One common use case of LCA is to explore heterogeneity and determine whether a sample is comprised of latent subgroups (Online Supplementary Materials A). In this use case, the focus is on class enumeration. Second, although LCA is often discussed as an exploratory analysis technique, it can be used in a confirmatory manner (Online Supplementary Materials B). Given its similarity to CFA, LCA it can be used to similar ends when a theory postulates the existence of a categorical latent variable: to assess whether the theoretical measurement model holds. LCA is also used to reduce high-dimensional data to just a few prototypes. While factor analysis is also used for dimension reduction, it only accounts for linear covariance between indicators. LCA, by contrast, can accommodate complex - but discrete - patterns. In this context, LCA can be used for pragmatic reasons without assuming the a existence of an underlying categorical latent variable. Consequently, the class solution should be treated as a set of prototypes that describe the sample. LCA is used for dimension reduction when, for example, there are many ordinal indicators (Online Supplementary Materials C), and in longitudinal research, when researchers identify different prototypical developmental trajectories and relate these to

auxiliary models (Online Supplementary Materials D). Another use case of LCA is to estimate individuals' unobserved class membership, for example, for diagnostic purposes in clinical contexts. A special case is the classification of new individuals that were not part of the sample used to estimate the model, which allows the LCA to be used as a diagnostic aid (Online Supplementary Materials B). Finally, LCA is also used to account for violations of normality assumptions (Online Supplementary Materials A). Special cases of this use case are zero-inflated and hurdle models, which account for censored data and extreme skew (Baughman, 2007). Their parameters differ across classes: one class accounts for the excess zeroes, and another class fits a different distribution to explain the remaining values, which can include zero. Such models are beyond the scope of this paper.

Contemporary Practices in Analysis and Reporting

The practices discussed here are rooted in existing recommendations for estimating (Nylund-Gibson & Choi, 2018; Van De Schoot, Sijbrandij, Winter, Depaoli, & Vermunt, 2017), and reporting (Masyn, 2013; Weller, Bowen, & Faubert, 2020) specific subtypes of LCA. These guidelines were updated, generalized to be relevant to all types of LCA, and implemented in `tidySEM`. They are summarized in the SMART-LCA Checklist: Standards for More Accuracy in Reporting of different Types of Latent Class Analysis (Online Supplementary Materials E)

Examining Observed Data. Examining data may reveal distributional idiosyncrasies and violations of assumptions. The function `descriptives(data)` provides extensive descriptive statistics, including the number of unique values, variance of continuous and categorical variables, missingness, skew and kurtosis. Density plots for continuous variables and bar charts for categorical ones can convey additional information. Verify that the observed distribution is consistent with the intended measurement level. For instance, continuous variables should have many unique values; if not, it may be better to model them as ordinal. Reporting sample descriptives is due diligence (Schreiber, 2017), but there are

two important limitations in the context of LCA. First, sample descriptives are not sufficient to check or reproduce LCA. Second, LCA assumes a heterogeneous population, which limits the utility of sample descriptives. It is thus also important to report class-specific descriptives. Note that normality tests have little use in LCA. A recent publication mistakenly stated that Gaussian mixture models assume normally distributed indicators (Spurk, Hirschi, Wang, Valero, & Kauffeld, 2020). The correct assumption is that observed distributions are a *mixture of normal distributions*, which is itself non-normal. If indicators are normally distributed, LCA may not be the appropriate technique.

Data Preprocessing. LCA can accommodate continuous, ordinal, binary, and nominal, indicators. Continuous variables should have type `numeric` or `integer` and an interval measurement level such that it is sensible to estimate their means and (co)variances. Ordered variables such as Likert scales, and binary variables like the presence or absence of symptoms should be converted to `mxFactor`. Nominal variables of type `factor` and `character` can be converted to binary dummy variables using `mx_dummies()`. Indicators on different scales may cause convergence issues. The reason for this is that the parameter space is high-dimensional with many potential local optima. If one indicator is on a much larger scale than another, the parameter space may be more extensively explored for that parameter. The problem can be resolved by transforming the indicators to place them on (roughly) similar scales, for example, by standardizing them using the `scale()` function. Model parameters can be transformed back to the original scale by reversing the transformation. Another consideration relates to the univariate distributions of indicators. For example, if indicators are zero-inflated and right-skewed, this must be taken into account either in preprocessing, or in model specification. In preprocessing, data can be transformed to reduce skew. After reducing skew, a simple LCA may fit the data well. In model specification, one can accommodates the distributional idiosyncrasies by, for example, allowing free variances across classes. This might result in one class with a near-zero mean and small variance and one class with an elevated mean and large variance. It is also

possible to specify separate models for each latent class, as in zero-inflated models, but this is beyond the scope of the present paper (Baughman, 2007).

Missing Data. Examining the pattern of missing data is due diligence (Van De Schoot et al., 2017). Three types of missingness are distinguished (Enders, 2022): random missingness (MCAR); missingness contingent on observed data (MAR); and missingness related to unobserved data (MNAR). Classic missingness tests assume normality; an assumption likely to be violated in LCA (Little, 1988). Instead, a non-parametric test can be used to determine whether missingness is MAR or not (Jamshidian & Jalal, 2010). In `tidySEM`, missingness is handled using Full Information Maximum Likelihood (FIML) estimation. As FIML assumes either MCAR or MAR, one would proceed with FIML regardless of the test outcome. FIML makes use of all available information (Enders, 2022) and performs on par with multiple imputation (Lee & Shi, 2021). Note that FIML estimates are only unbiased if all predictors of missingness are included in the model. Multiple imputation is less suitable for LCA because many imputation methods assume normality, whereas LCA effectively assumes non-normality (i.e., it assumes a mixture of normal distributions). Moreover, aggregating model fit indices and parameter estimates across imputed datasets is nontrivial in LCA. There is no universally best approach to handling missing data, but given these considerations, our recommendation is to report proportions of missingness per variable using `descriptives()`, test for MAR using `mice::mcar()`, and use FIML estimation.

Model Specification. LCA can be based on any parametric model whose parameters are appropriate for the indicators' measurement level. Parameters can be freely estimated, constrained across classes, or fixed (e.g., to zero). Continuous indicators are parametrized in terms of means, variances, and (co)variances. Ordinal indicators are parametrized using thresholds corresponding to quantiles of a standard normal distribution. A binary indicator has a single threshold that distinguishes the two response categories. If responses are distributed 50/50, the threshold would be $t_1 = 0.00$; if responses are

distributed 60/40, then the resulting threshold would be $t_1 = 0.25$. The wrapper functions `mx_profiles()` and `mx_lca()` construct LCA models for continuous and ordinal indicators respectively. More complex models can be specified in `lavaan` syntax using `mx_mixture()`. A common pitfall is treating ordinal data as continuous. Although a widespread rule of thumb suggests that ordinal scales with 7+ categories can be treated as continuous (Rhemtulla, Brosseau-Liard, & Savalei, 2012), doing so can cause problems in LCA. To understand why, imagine a 7-class LPA with a single 7-category indicator. Each class mean would describe a single response category, and class-specific variance components would be empirically underidentified.

Prior literature has emphasized considering alternative model specifications (Van De Schoot et al., 2017). What has remained underemphasized is that alternative model specifications play a different role in exploratory versus confirmatory LCA. Exploratory LCA typically involves a search for the best model along several model specifications and numbers of classes (Nylund, Asparouhov, & Muthén, 2007). A pitfall in this context is omitting the one-class solution; if it has relatively good fit, LCA might not be appropriate. Confirmatory LCA, by contrast, focuses on one or more hypothesized (and optionally preregistered) models. Alternative model specifications still play a role because LCA lacks objective fit indices. One way to test the theoretical model is to demonstrate that its fit compares favorably to alternative models, such as a one-class model, models with a few more or less classes, or competing theoretical models. If the hypothesized model fits much better, this provides evidence for the theory. If it fits much worse, this provides evidence against the theory. If fit differences are small, the theoretical model may still be preferred, as theory is a relevant consideration in model selection (Jung & Wickrama, 2008).

Maximum Number of Classes. One aspect of model specification is determining the maximum number of classes to be considered. The maximum may be determined on theoretical grounds, limited by computational resources, or informed by the data. Above a certain number of classes, convergence problems may occur, and classes may become very

small. Low class counts may be a reason to eliminate more complex models due to the risk of overfitting (Hastie, Tibshirani, & Friedman, 2009). Overfitting occurs when a model has many parameters relative to the number of observations, causing it to capture idiosyncratic noise in the sample and limiting its generalizability to new samples. In LCA, it is important to consider not only the overall ratio of observations to parameters, but also the class-specific ratio. Small classes may be locally non-identified if they comprise too few observations to estimate class-specific parameters (Depaoli, 2013).

Estimation and Convergence. A unique challenge in LCA is that model parameters and class membership must be estimated simultaneously. Participants contribute to the estimation of parameters in all classes, and their contributions to each class are weighted by their probability of belonging to that class - which is in turn computed based on the class parameters. The resulting chicken-and-egg problem is resolved by the expectation-maximization algorithm: Estimation proceeds iteratively, alternating between calculating posterior class probabilities based on the current parameter estimates, then using maximum likelihood (ML) to re-estimate the parameters using the new class probabilities as weights, and so on, until both have converged. ML finds the combination of parameter values that minimizes a measure of discrepancy between the model and the observed data: the minus two log likelihood, $-2LL$. To understand ML, imagine we are estimating two parameters: the class-specific means μ_c on a continuous indicator for two classes. Imagine a three-dimensional landscape where the X and Y dimensions represent potential values of the class means, $X = \mu_1$ and $Y = \mu_2$, and the Z-dimension represents the value of the $-2LL$. The optimizer must find the deepest "valley" in this landscape: the combination of μ_1 and μ_2 that minimizes the $-2LL$. The ML optimizer behaves somewhat like a marble, dropped at a random point in this landscape: It rolls towards the nearest valley (a combination of values for μ_1 and μ_2 that has a low $-2LL$). One challenge is that LCA models often have very many parameters, which means that the landscape is not three- but very high-dimensional, with many valleys and sparse data to provide information about which

valley is deepest. If the marble rolls into one valley, it will settle there and not climb out again. One risk is that the optimizer gets stuck in a shallower valley (a "local optimum"), and never discovers the deepest valley (the "global optimum"). One solution to this problem is to drop many marbles at random places and make sure that several marbles find the same valley. This is the "random starts" approach implemented in, e.g., Mplus.

Random starts are computationally expensive, inefficient because start values may be far from an optimum, and there is a risk that the global optimum is never found if all starting values are close to a specific local optimum. These challenges are overcome by the global optimizer simulated annealing (SA, Corana, Marchesi, Martini, & Ridella, 1987). At each iteration, SA considers a "destination" in the landscape, and compares its $-2LL$ to the current one. If the destination $-2LL$ is lower, it moves there. However, if the destination $-2LL$ is higher, it still moves there occasionally. This allows it to escape local optima. Think of this as occasionally flicking a marble to see if it rolls back into the current valley or finds another deeper one. Since SA is a global optimizer, its outcome is independent of starting values (Corana et al., 1987). If we assume that the classes have different means, we can use nonparametric clustering and multigroup analysis to determine reasonable start values (Biernacki, Celeux, & Govaert, 2003). `tidySEM` uses K-means clustering for complete data (Hartigan & Wong, 1979), and hierarchical clustering when there are missing values. SA is then used to find the global optimum, followed by a short run of ML to obtain the asymptotic covariance matrix.

In practice, models do not always converge. Convergence problems may be indicated by errors and warnings from the estimator, or they may be suspected based on improbable parameter values or changing results when the analyses are reproduced. One way to address nonconvergence and local optima is by re-estimating the model while permuting the starting values. The `OpenMx` package contains a family of functions for this purpose: `mxTryHard()` and `mxTryHardWideSearch()` for continuous indicators, and `mxTryHardOrdinal()` for ordinal indicators. Alternatively, starting values may be specified manually specified using

`OpenMx::omxSetParameters()` before re-running the model. Convergence problems may also be caused by small samples or poor data quality. In these cases, the only solution is to collect more, or higher quality data. Another potential reason for nonconvergence is incorrect model specification. The model may be too complex for the data, or conversely, too simple to capture important patterns. For example, consider LCA with multiple ordinal indicators. If certain combinations of responses rarely occur together, this can result in extreme conditional item probabilities (exactly 0 or 1). A solution may be to merge response categories with very few responses, or to simplify the model by estimating fewer classes. Similar problems occur with continuous indicators, for example with zero-inflated distributions (see Data Preprocessing). To account for the excess zeroes, one class will have a mean of zero and a very small, or zero, standard deviation. Constraining the standard deviations across classes would likely result in nonconvergence.

Model Fit Indices. Due to the lack of absolute fit indices for LCA, relative measures of model fit are used. These so-called information criteria (ICs) are computed from the most basic fit measure, $-2LL$; see `table_fit()`. The lower its value, the better the model fits the data. However, there is a balance between how well a model fits the data at hand, and how well it generalizes to new data (Hastie et al., 2009). To curtail such overfitting, ICs penalize model complexity. The Akaike Information Criterion (AIC) is the original information criterion (Akaike, 1974). It is computed as $-2LL + 2p$, where p is the number of parameters. The Bayesian Information Criterion (BIC) multiplies the penalty for complexity with a function of the sample size, $p \ln(n)$, such that complex models are penalized more in smaller samples (Schwarz, 1978), which incur greater risk of overfitting (Hastie et al., 2009). The BIC may be a sensible default IC for class enumeration as it is widely known and performs well across simulation studies (Masyn, 2013; Nylund-Gibson & Choi, 2018; Vermunt, 2023). Finally, the so-called sample size adjusted BIC (saBIC), $\ln(\frac{n+2}{24})$, implements a penalty for sample size based on the smallest number of pieces of information necessary to reproduce the data (Rissanen, 1983; Sclove, 1987). The saBIC

performs well for homogeneous clusters in small samples but selects too many classes when the number of the number of classes and the sample size are large (Chen, Luo, Palardy, Glaman, & McEnturff, 2017; Tein, Coxe, & Cham, 2013).

In class enumeration, the model with the lowest IC value is often preferred, as it best balances fit and complexity. An informal way to further penalize complexity is to use a scree plot, `plot()`, to determine the inflection point at which additional classes only marginally decrease the ICs (Nylund-Gibson & Choi, 2018). Another way to compare models is using IC-weights, `ic_weights()`, which indicate the relative support the data provides for each model in a set (Wagenmakers & Farrell, 2004). This is particularly useful for confirmatory LCA, as it allows one to quantify support for the theoretical model as a percentage. Note that IC-weights (strongly) favor the model with the lowest IC value. ICs may contradict each other, so it is important to consider how to reconcile them. One option is to preregister a specific IC before seeing the data. Another option is to select the most parsimonious model that has best fit according to any IC. Another option is to integrate information from multiple fit indices (Akogul & Erisoglu, 2016).

Likelihood Ratio Tests. A common practice is to use Likelihood Ratio tests (LRT) for class enumeration. In particular, the Lo-Mendell-Rubin LRT (LMR) is widely used to test the significance of the difference between k- and $k-1$-class models (Lo, Mendell, & Rubin, 2001). The popularity of this test notwithstanding, an insurmountable limitation is that it assumes regularity conditions that are generally violated in LCA (especially when mixture components are non-normal) (Jeffries, 2003). Moreover, recent work confirms that the LMR does not yield uniformly distributed p-values under the null hypothesis, and that Mplus' implementation differs from the validated method and performs worse (Vermunt, 2023). The bootstrapped LRT overcomes some of these limitations (BLRT, Nylund et al., 2007; Tein et al., 2013). It simulates data from the $k-1$ class model b times (e.g., 1000), and calculates the LRT for k versus $k-1$ class models in each simulated dataset. Its p-value is the proportion of b in which the test statistic exceeds the original LR. The BLRT is

computationally expensive and sensitive to model misspecification, which limits its use in exploratory LCA (Nylund et al., 2007). For confirmatory analyses, however, the BLRT should be preferred over the LMR. All LRTs thus have some limitations, and in many cases the BIC may be preferred for class enumeration (Vermunt, 2023). One questionable practice pertaining to LRT is stepwise class enumeration, whereby additional models are estimated until a non-significant LRT is observed (Spurk et al., 2020). This is problematic because of the aforementioned limitations of LRTs, and because stopping at the first non-significant LRT could prevent the selection of a better model with an even higher number of classes.

Classification Diagnostics. Classification diagnostics assess whether classes are distinct and clearly separable. As these diagnostics are not fit indices, they should not be used for model selection. A model can fit the data well and still exhibit poor class separability (Masyn, 2013). However, classification diagnostics can be used to restrict the search space of acceptable models. For example, when the analysis goal is to interpret properties of classes or their members, it makes sense to consider only models with clear class separation (Nylund-Gibson & Choi, 2018).

All classification diagnostics are derived from the posterior classification probabilities P. This $n \times k$ matrix contains the probabilities for every individual i of belonging to latent classes $1 \dots k$. When classification accuracy is high and classes are distinct, each individual has a high posterior class probability for one class, and low for all others. This matrix is obtained by running `class_prob(res, type = "individual")`. Note that one column is added with each individual's most likely class membership M, based on the highest posterior class probability. The matrix P can be summarized in different ways. For example, the estimated frequency table of the latent categorical variable F_P is obtained by taking the column sums of P, see `class_prob(res, type = "sum.posterior")`. This table takes classification error into account, so individuals can contribute partially to multiple classes. Second, the frequency table of the variable M, F_M, is obtained by running `class_prob(res, "sum.mostlikely")`. Unlike F_P, F_M ignores classification error. If every

observation is classified with perfect accuracy, F_M and F_P are identical.

One important classification diagnostic is the size of the smallest class, derived from F_M. It is included in the output of `table_fit()` as a percentage of the sample in the column `n_min`. Existing rules of thumb have discussed the smallest class size in terms of percentage of the total sample size (Jung & Wickrama, 2008). The main concern here is that the smallest class should have a unique and meaningful substantive interpretation. This does not, however, address the concern that small classes may comprise too few observations to accurately estimate class parameters, a phenomenon known as local non-identification (Depaoli, 2013). Thus, a better rule of thumb might be to ascertain that the ratio of observations per estimated parameter is sufficiently large, both across and within classes. Ratios of 10 or 20 observations per parameter have been suggested in the literature (Jackson, 2003; Kline, 2016). If the smallest class has too few observations per parameter, or its substantive interpretation is redundant with another class, a simpler solution with fewer classes or a simpler class-specific model is preferable.

To assess classification accuracy, we turn to the third diagnostic table: the average posterior probabilities by most likely class membership, $P_{p_{(M=m)}}$, obtained via `class_prob(res, type = "avg.mostlikely")`. This table gives the column means of P for the k subsets of observations with most likely class of $M = 1 \ldots k$. Each row gives the mean probability of being assigned to each class for participants assigned to one particular class according to M. The diagonal of this matrix represents the certainty of class assignment. Some guidelines have suggested that values on the diagonal should exceed 0.7 (Masyn, 2013). These guidelines should not be considered as strict cutoffs; rather, classification quality should be interpreted substantively.

The table of classification probabilities for the most likely latent class M by latent class, $P_{p_{(M=m|C=c)}}$ is similar to $P_{p_{(M=m)}}$, except that it accounts for classification uncertainty. If C is the true class of an observation, which is imperfectly measured, and M is the most

likely class, then this table shows the probability of an observation being assigned to class m in M, given that its true class is c in C, or $P(M = m|C = c)$. Values on the diagonal indicate the reliability with which each level of the categorical latent variable is measured by M, and off-diagonal elements can be conceptualized as measurement error. As a summary of classification accuracy, the minimum and maximum of the diagonal are given by default as a part of the `table_fit()` output (`prob_min` and `prob_max`). Both probabilities should be high, as this means that all classes are reliably classified. A low minimum probability indicates at least one class with low classification accuracy.

The entropy criterion summarizes class separability in a single statistic (Celeux & Soromenho, 1996). In physics, high entropy refers to maximum randomness or uncertainty, and low entropy corresponds to a strict arrangement of the units of study. Applied to LCA, high entropy means that classes are completely indistinguishable, and low entropy means that classes are fully separated. Following the Mplus convention, `tidySEM` reports 1-entropy, reversing its interpretation: 0 means the model classification is no better than chance, and 1 means that every individual has a near-one probability of being in one particular class. If the goal is to identify clearly distinct classes, higher entropy values are preferred. There are rules of thumb for entropy because its value tends to be lower when the number of samples, indicators, or latent classes is higher.

Labelling Classes. Class names should be chosen to accurately reflect theoretically relevant class characteristics. These names should aid interpretation and serve as a shorthand throughout the manuscript. For example, one study used a dual-trajectory LCGA of cognitive and affective empathy development with gender as a covariate. Although this model has dozens of parameters, the resulting classes were labeled low, average, and high empathy because level differences were the most salient difference between classes. According to the *naming fallacy*, overly simplistic and general class names can be misleading (Weller et al., 2020). For example, it would be misleading to call one group “low empathy” and another group “medium empathy” if the empathy differences were non-significant, or if other

differences, like trajectory shape, were more substantial. To avoid the naming fallacy, comprehensively report model parameters and emphasize that class names are just a shorthand. When naming classes, be aware that LCA models are identified up to the class order. This order may change if the analysis is replicated, a phenomenon known as label switching. To ensure that classes stay in the same order upon replication, set a random seed before fitting the model by using `set.seed()`. To set a specific class order, use the function `mx_switch_labels()`. By default, it orders classes from largest to smallest class count - but classes can also be ordered by their values on a specific parameter (e.g., the mean intercept in a latent class growth model). Label switching is a larger problem in Bayesian LCA, but this is beyond the scope of this paper.

Reporting: Transparency and Reproducibility. Journals, funding bodies and professional organizations increasingly require open and reproducible research. Many have adopted the TOP guidelines, which recommend comprehensive citation of literature, software, and data sources; sharing data and analysis code, and pre-registration (Nosek et al., 2016). The FAIR principles set the standard for how research output should be shared (see Lamprecht et al., 2020). The Workflow for Open Reproducible Code in Science describes how to create a reproducible research archive consistent with these guidelinges, and the `worcs` R-package automates many of the steps involved (C. J. Van Lissa et al., 2021). This workflow uses a dynamic document that automatically generates analysis results from code, which is version controlled to ensure a complete historical record of the project, and tracks all software dependencies to ensure reproducible results. Using open-source software is a necessary condition for reproducibility. By combining `tidySEM` and `worcs`, it is possible to conduct a fully reproducible LCA analysis compliant with established standards. Data sharing is especially important for LCA compared to other methods, because LCA requires individual participant data to be reproduced. If the original data cannot be shared, `worcs` automatically generates synthetic data to ensure computational reproducibility. However, generating synthetic data from the final model using `mxGenerateData()` will provide a

better basis for reproducing LCA results.

Reporting: Preregistration. LCA analyses inherently involve some subjectivity, because of the potential for disagreement between criteria for class enumeration and the lack of objective fit indices. Preregistering the criteria for class enumeration and determining the maximum number of classes eliminates concerns that these were cherry picked after seeing the results. The `worcs` package facilitates creating a preregistration from standard templates. More interestingly, it facilitates creating a *Preregistration As Code* (PAC): A draft paper with reproducible analysis section based on fake (synthetic or simulated) data (Peikert, Van Lissa, & Brandmaier, 2021). In PAC, the class enumeration criteria can be hard-coded into the analysis section, removing any ambiguity. Once real data are accessed or collected, the analysis is simply reproduced, and the results section is automatically updated. PAC leaves less room for ambiguity than a conventional preregistration because the analysis plan is not merely described but implemented in code. Moreover, since a PAC is the first draft of a paper, writing it saves time compared to filling out preregistration forms. For an example of PAC, see C. J. Van Lissa (2022a).

Reporting: Model Fit. The function `table_fit()` reports an overview of fit indices, customized for LCA. First, report the effective sample size, which should be the same for all models in order to compare ICs. Second, report the number of parameters for each model, and clearly explicate which parameters are estimated across and within classes. For a full overview of the parameters, use the `coef()` or `table_results()` functions. Third, report the relative fit of different solutions. By default, `table_fit()` includes the $-2LL$ and three ICs: AIC, BIC, and saBIC. Fourth, report classification diagnostics. Of particular interest is the range of the diagonal of the most likely class membership by true class membership, which indicates classification reliability (`prob_min` and `prob_max`). The entropy is also of interest as an overall measure of class separation. Finally, the range of proportions of cases assigned to each class is important (`n_min` and `n_max`); the minimum proportion of cases may indicate classes that are too small to be locally identified or

practically relevant (see Depaoli, 2013). Finally, a test like the BLRT can be reported if it is part of the class enumeration strategy, using `BLRT()`.

Reporting: Results. LCA results include class proportions, `class_prob(res, "sum.posterior")`, obtained by standardizing the class weights. A k class model has $k-1$ free weights; one weight is redundant and thus constrained. Unstandardized weights and other parameters are obtained via `table_results(res)`. Note that ordinal thresholds are easier to interpret when converted to probabilities, using `table_prob()`. It is good practice to report point estimates, standard errors, and confidence intervals.

Reporting: Inference. The results table includes p-values for the hypothesis that the parameter is equal to zero. Whether this is informative depends on the research questions and variable scale. For mean-centered indicators, the p-values of class means indicate whether they differ significantly from the sample average. Other null hypotheses can be tested using the standard errors. Although a recent article suggested that robust standard errors should be used when indicators are not normally distributed (Spurk et al., 2020), we found no research supporting this claim. Informative hypotheses can be tested using `wald_test()` (see Caspar J. Van Lissa et al., 2020). In confirmatory LCA, a theoretical model can be tested by comparing model parameters against their hypothesized values. As LCA models are often highly parametrized, it is important to control the risk of Type I errors, or at least reflect on it (Goeman & Solari, 2011).

Reporting: Visualization. In an informal review of LCA visualization, we found that most graphics focus on model parameters, displaying class means as points connected by lines, with point, shape and line type indicating class membership. This approach has some limitations: These plots omit parameter uncertainty, do not illustrate how well the model describes the data, and the lines suggest a non-existent continuous connection between items. Plots in `tidySEM` are based on best practices according to the Grammar of Graphics (Wilkinson, 2005). They visualize raw data, which enables assessing class separability and how well the data are captured by the model. Color plots indicate class membership by color,

and publication-friendly black and white plots instead use line type and point shape. Point estimates and raw data are visualized using points. Although these points can be connected by lines in `plot_profiles()`, this is discouraged because the lines do not convey any additional information. More salient elements are error bars for standard errors and standard deviations. Density plots visualize univariate observed distributions in `plot_density()`, and ellipses visualize (co)variances in `plot_bivariate()`. When using plots to aid class enumeration, competing models are juxtaposed. As `tidySEM` plots are implemented in `ggplot2` and remain class compatible, they can be further customized. One remaining limitation is that further work remains to be done to implement recommended practices for plots of ordinal indicators (`plot_prob()`) and for longitudinal LCA (`plot_growth()`).

Follow-up Analyses. Researchers often wish to relate class membership to auxiliary variables, or even auxiliary models. It is important to keep in mind that most likely class membership M is subject to classification error (Vermunt, 2010a). Treating class membership as an observed variable disregards this error, resulting in biased models, except when class separation is very high. It is therefore important to account for classification error in follow-up analyses. The BCH method, available via `BCH()`, compares auxiliary variables or arbitrary auxiliary models across classes while accounting for classification error (Bolck, Croon, & Hagenaars, 2004).

Conclusions

This paper discussed types of LCA and their use cases, illustrated via tutorials in the Online Supplementary Materials and package vignettes. Expanding and updating existing guidelines, we recommend practices for estimating and reporting LCAs. These recommendations are summarized in the SMART-LCA checklist to guide authors, readers, and reviewers. We implemented LCA in the free open-source R-package `tidySEM`, paying particular attention to the practices discussed in this paper. The package's user-friendly interface brings advanced LCA within reach of novice users, while its embedding within the

extensible `OpenMx` framework ensures its relevance for advanced users. We encourage contributions that further enhance its capabilities. Our aim is to democratize access to cutting edge latent class analysis and incentivize further methodological and applied research in this area.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19*(6), 716–723. https://doi.org/10.1109/TAC.1974.1100705

Akogul, S., & Erisoglu, M. (2016). A comparison of information criteria in clustering based on mixture of multivariate normal distributions. *Mathematical and Computational Applications*, *21*(3), 34. https://doi.org/10.3390/mca21030034

Baughman, A. L. (2007). Mixture model framework facilitates understanding of zero-inflated and hurdle models for count data. *Journal of Biopharmaceutical Statistics*, *17*(5), 943–946. https://doi.org/10.1080/10543400701514098

Biernacki, C., Celeux, G., & Govaert, G. (2003). Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate gaussian mixture models. *Computational Statistics & Data Analysis*, *41*(3), 561–575. https://doi.org/10.1016/S0167-9473(02)00163-9

Bolck, A., Croon, M., & Hagenaars, J. (2004). Estimating latent structure models with categorical variables: One-step versus three-step estimators. *Political Analysis*, *12*(1), 3–27. https://doi.org/10.1093/pan/mph001

Celeux, G., & Soromenho, G. (1996). An entropy criterion for assessing the number of clusters in a mixture model. *Journal of Classification*, *13*(2), 195–212. https://doi.org/10.1007/BF01246098

Chen, Q., Luo, W., Palardy, G. J., Glaman, R., & McEnturff, A. (2017). The efficacy of common fit indices for enumerating classes in growth mixture models when nested data structure is ignored: A monte carlo study. *SAGE Open*, *7*(1), 215824401770045. https://doi.org/10.1177/2158244017700459

Collins, L. M., & Lanza, S. T. (2009). *Latent class and latent transition analysis: With applications in the social, behavioral, and health sciences*. John Wiley & Sons.

Corana, A., Marchesi, M., Martini, C., & Ridella, S. (1987). Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm—corrigenda

for this article is available here. *ACM Transactions on Mathematical Software*, *13*(3), 262–280. https://doi.org/10.1145/29380.29864

Depaoli, S. (2013). Mixture class recovery in GMM under varying degrees of class separation: Frequentist versus bayesian estimation. *Psychological Methods*, *18*(2), 186–219. https://doi.org/10.1037/a0031609

Enders, C. K. (2022). *Applied missing data analysis*. Guilford Publications.

Figueiredo, M. A. T., & Jain, A. K. (2002). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(3), 381–396. https://doi.org/10.1109/34.990138

Goeman, J. J., & Solari, A. (2011). Multiple Testing for Exploratory Research. *Statistical Science*, *26*(4). https://doi.org/10.1214/11-STS356

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*(1), 100–108. Retrieved from http://www.jstor.org/stable/2346830

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (Second). New York: Springer.

Hennig, C., Meila, M., Murtagh, F., & Rocci, R. (2015). *Handbook of cluster analysis*. 28.

Jackson, D. L. (2003). Revisiting Sample Size and Number of Parameter Estimates: Some Support for the N:q Hypothesis. *Structural Equation Modeling: A Multidisciplinary Journal*, *10*(1), 128–141. https://doi.org/10.1207/S15328007SEM1001_6

Jamshidian, M., & Jalal, S. (2010). Tests of homoscedasticity, normality, and missing completely at random for incomplete multivariate data. *Psychometrika*, *75*(4), 649–674. https://doi.org/10.1007/s11336-010-9175-3

Jeffries, N. O. (2003). A Note on 'Testing the Number of Components in a Normal Mixture'. *Biometrika*, *90*(4), 991–994. Retrieved from https://www.jstor.org/stable/30042105

Jung, T., & Wickrama, K. A. S. (2008). An introduction to latent class growth analysis and growth mixture modeling. *Social and Personality Psychology Compass*, *2*(1), 302–317.

https://doi.org/10.1111/j.1751-9004.2007.00054.x

Kline, R. B. (2016). *Principles and Practice of Structural Equation Modeling* (Fourth). New York: The Guilford Press.

Lamprecht, AL., Garcia, L., Kuzak, M., Martinez, C., Arcila, R., Martin Del Pico, E., … Capella-Gutierrez, S. (2020). Towards FAIR principles for research software. *Data Science*, *3*(1), 37–59. https://doi.org/10.3233/DS-190026

Lee, T., & Shi, D. (2021). A comparison of full information maximum likelihood and multiple imputation in structural equation modeling with missing data. *Psychological Methods*, No Pagination Specified–No Pagination Specified. https://doi.org/10.1037/met0000381

Little, R. J. A. (1988). A test of missing completely at random for multivariate data with missing values. *Journal of the American Statistical Association*, *83*(404), pp. 1198–1202. https://doi.org/10.2307/2290157

Lo, Y., Mendell, N. R., & Rubin, D. B. (2001). Testing the number of components in a normal mixture. *Biometrika*, *88*(3), 767–778. https://doi.org/10.1093/biomet/88.3.767

Masyn, K. E. (2013). Latent class analysis and finite mixture modeling. In *The oxford handbook of quantitative methods*: *Vol. 2: Statistical Analysis* (p. 551). Oxford University Press.

Molenaar, P. C. M., & Eye, A. von. (1994). On the arbitrary nature of latent variables. In *Latent variables analysis: Applications for developmental research* (pp. 226–242). Thousand Oaks, CA, US: Sage Publications, Inc.

Muthén, L. K., & Muthén, B. O. (1998). *Mplus user's guide*. Los Angeles, CA: Muthén & Muthén.

Neale, M. C., Hunter, M. D., Pritikin, J. N., Zahery, M., Brick, T. R., Kirkpatrick, R. M., … Boker, S. M. (2016). OpenMx 2.0: Extended structural equation and statistical modeling. *Psychometrika*, *81*(2), 535–549. https://doi.org/10.1007/s11336-014-9435-8

Nosek, B. A., Alter, G., Banks, G. C., Borsboom, D., Bowman, S., Breckler, S., … DeHaven, A. C. (2016). *Transparency and openness promotion (TOP) guidelines*. OSF

Preprints. https://doi.org/10.31219/osf.io/vj54c

Nylund, K. L., Asparouhov, T., & Muthén, B. O. (2007). Deciding on the number of classes in latent class analysis and growth mixture modeling: A monte carlo simulation study. *Structural Equation Modeling: A Multidisciplinary Journal*, *14*(4), 535–569. https://doi.org/10.1080/10705510701575396

Nylund-Gibson, K., & Choi, A. Y. (2018). Ten frequently asked questions about latent class analysis. *Translational Issues in Psychological Science*, *4*(4), 440–461. https://doi.org/10.1037/tps0000176

Peikert, A., Van Lissa, C. J., & Brandmaier, A. M. (2021). Reproducible research in r: A tutorial on how to do the same thing more than once. *Psych*, *3*(4), 836–867. https://doi.org/10.3390/psych3040053

Rhemtulla, M., Brosseau-Liard, P. É., & Savalei, V. (2012). When can categorical variables be treated as continuous? A comparison of robust continuous and categorical SEM estimation methods under suboptimal conditions. *Psychological Methods*, *17*, 354–373. https://doi.org/10.1037/a0029315

Rissanen, J. (1983). A Universal Prior for Integers and Estimation by Minimum Description Length. *The Annals of Statistics*, *11*(2), 416–431. https://doi.org/10.1214/aos/1176346150

Rosseel, Y. (2012). Lavaan: An r package for structural equation modeling. *Journal of Statistical Software*, *48*, 1–36. https://doi.org/10.18637/jss.v048.i02

Schreiber, J. B. (2017). Latent class analysis: An example for reporting results. *Research in Social and Administrative Pharmacy*, *13*(6), 1196–1201. https://doi.org/10.1016/j.sapharm.2016.11.011

Schwarz, G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, *6*(2), 461–464. https://doi.org/10.1214/aos/1176344136

Sclove, S. L. (1987). Application of model-selection criteria to some problems in multivariate analysis. *Psychometrika*, *52*(3), 333–343. https://doi.org/10.1007/BF02294360

Scrucca, L., Fop, M., Murphy, T. B., & Raftery, A. E. (2016). Mclust 5: Clustering, classification and density estimation using gaussian finite mixture models. *The R Journal*, *8*(1), 289–317.

Spurk, D., Hirschi, A., Wang, M., Valero, D., & Kauffeld, S. (2020). Latent profile analysis: A review and "how to" guide of its application within vocational behavior research. *Journal of Vocational Behavior*, *120*, 103445. https://doi.org/10.1016/j.jvb.2020.103445

Tein, JY., Coxe, S., & Cham, H. (2013). Statistical power to detect the correct number of classes in latent profile analysis. *Structural Equation Modeling: A Multidisciplinary Journal*, *20*(4), 640–657. https://doi.org/10.1080/10705511.2013.824781

Van De Schoot, R., Sijbrandij, M., Winter, S. D., Depaoli, S., & Vermunt, J. K. (2017). The GRoLTS-checklist: Guidelines for reporting on latent trajectory studies. *Structural Equation Modeling: A Multidisciplinary Journal*, *24*(3), 451–467. https://doi.org/10.1080/10705511.2016.1247646

Van Lissa, C. J. (2022a). Complementing preregistered confirmatory analyses with rigorous, reproducible exploration using machine learning. *Religion, Brain & Behavior*, *0*(0), 1–5. https://doi.org/10.1080/2153599X.2022.2070254

Van Lissa, C. J. (2022b). *tidySEM: Tidy structural equation modeling*. Retrieved from www.github.com/cjvanlissa/tidySEM

Van Lissa, C. J., Brandmaier, A. M., Brinkman, L., Lamprecht, AL., Peikert, A., Struiksma, M. E., & Vreede, B. M. I. (2021). WORCS: A workflow for open reproducible code in science. *Data Science*, *4*(1), 29–49. https://doi.org/10.3233/DS-210031

Van Lissa, Caspar J., Gu, X., Mulder, J., Rosseel, Y., Zundert, C. V., & Hoijtink, H. (2020). Teacher's Corner: Evaluating Informative Hypotheses Using the Bayes Factor in Structural Equation Models. *Structural Equation Modeling: A Multidisciplinary Journal*, *0*(0), 1–10. https://doi.org/10.1080/10705511.2020.1745644

Vermunt, J. K. (2010a). Latent Class Modeling with Covariates: Two Improved Three-Step Approaches. *Political Analysis*, *18*(4), 450–469. https://doi.org/10.1093/pan/mpq025

Vermunt, J. K. (2010b). Longitudinal research using mixture models. In K. van Montfort, J. H. L. Oud, & A. Satorra (Eds.), *Longitudinal research with latent variables* (pp. 119–152). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-642-11760-2_4

Vermunt, J. K. (2023). *The Vuong-Lo-Mendell-Rubin test for latent class and latent profile analysis: A note on the different implementations in Mplus and LatentGOLD*. Private website.

Vermunt, J. K., & Magidson, J. (2000). Latent GOLD 2.0 user's guide. *Statistical Innovations Inc.*

Vermunt, J. K., Magidson, J., Lewis-Beck, M., Bryman, A., Liao, T. F., & Department of Methodology and Statistics. (2004). Latent class analysis. In *The sage encyclopedia of social sciences research methods* (pp. 549–553). Sage. Retrieved from https://research.tilburguniversity.edu/en/publications/0caedd00-27c1-42bd-bb4e-d1dcb0864956

Wagenmakers, EJ., & Farrell, S. (2004). AIC model selection using akaike weights. *Psychonomic Bulletin & Review*, *11*(1), 192–196. https://doi.org/10.3758/BF03206482

Weller, B. E., Bowen, N. K., & Faubert, S. J. (2020). Latent class analysis: A guide to best practice. *Journal of Black Psychology*, *46*(4), 287–311. https://doi.org/10.1177/0095798420930932

Wilkinson, L. (2005). *The grammar of graphics*. New York: Springer-Verlag. https://doi.org/10.1007/0-387-28695-0