Best Practices in Mixture Modeling using Free Open Source Software

Caspar J. van Lissa[1,2]

[1] Utrecht University, Methodology & Statistics

[2] Open Science Community Utrecht

Author Note

Correspondence concerning this article should be addressed to Caspar J. van Lissa, Padualaan 14, 3584CH Utrecht, The Netherlands. E-mail: c.j.vanlissa@gmail.com

Abstract

Latent class analysis is a popular technique for identifying groups in data based on a parametric model. Examples of this technique are known as mixture models, latent profile analysis, latent class analysis, growth mixture modeling, and latent class growth analysis. Despite the popularity of this technique, there is limited guidance with respect to best practices in estimating and reporting mixture models. Moreover, although user-friendly interfaces for advanced mixture modeling have long been available in commercial software packages, open source alternatives have remained somewhat inaccessible. This tutorial describes best practices for the estimation and reporting of latent class analysis, using free and open source software in R. To this end, this tutorial introduces new functionality for estimating and reporting mixture mixture models in the `tidySEM` R-package. These functions rely on estimation using the OpenMx R-package.

Keywords: keywords

Word count: X

Best Practices in Mixture Modeling using Free Open Source Software

Latent class analysis has become quite popular across scientific fields, and under a number of different names. The purpose of this technique is to estimate unobserved group membership based on a parametric model of one or more observed indicators of group membership. Despite the popularity of the method, there is a noted absence of guidelines for estimating and reporting latent class analyses []. This complicates manuscript review and assessment of the quality of published studies, and introduces a risk of misapplications of the technique. The present paper seeks to address this gap in the literature by suggesting updated guidelines for estimation and reporting on latent class analysis, based on best practice.

Throughout this paper, examples make use of free, open source software for latent class analysis in R.

Defining latent class analysis

Latent class analysis can be understood as a method for estimating unobserved groups based on a parametric model of observed indicators of group membership. The concept of latent class analysis can be understood in different ways. Generally speaking, it can be said that a mixture model assumes that the study population is composed of K subpopulations, or classes. It further assumes that the observed data are a mixture of data generated by class-specific models. The simplest univariate "model" is a normal distribution, which can be described with two parameters: the mean and the variance. Commonly, the same model is estimated accross all classes, but with different parameters for each class (i.e., class-specific means and variances). Mixture modeling then estimates both the parameters for each class, and the probability for each that an individual belongs to each class.

As an illustrative example, imagine that a detective wants to know if it would be possible to use mixture modeling to identify the sex of a suspect, based on footprints found at the crime scene. To test the feasibility of this approach, the detective records the shoe sizes and sex of 100 volunteers. The resulting observed data look like this:

The distribution is evidently bimodal, which bodes well for the intended mixture model. In this case, the number of classes is known a-priori. When estimating a two-class mixture model, the detective observes that the model estimates the mean shoesize of the two groups are equal to 7.25 and 9.22, which is close to the true means of the two groups, namely 9.04 and 6.93. When tabulating estimated group membership against observed (known) group membership, it can be seen that women are classified with a high degree of accuracy, but men are not:

One way to is to conceptualize latent class analysis is by analogy to a measurement model in structural equation modeling. A mixture model is like confirmatory factor analysis, except that the continuous latent variable is substituted with a categorical latent variable. One difference between the two techniques is that factor analysis can be considered as a way to group observed *variables* into latent constructs, with factor loadings indicating which items belong are most indicative of a construct. By contrast, mixture modeling groups *individuals* into classes (see **nylund-gibsonTenFrequentlyAsked2018?**). In line with this distinction, latent class analysis is sometimes referred to as a "person-centered" technique, and factor analysis as a "variable-centered" technique.

When the focus is on the model parameters in each group, then latent class analysis can be thought of as similar to a multi-group structural equation model. The main distinction is that group membership is not known a-priori, but is instead estimated - with measurement error - based on the data. Whereas in a multigroup model, the data are split by group and treated as independent samples, in a mixture model, all cases contribute to the estimation of all parameters in all groups. The relative contribution of each case to the parameters of each group is determined by that case's posterior probability of belonging to that group.

When the focus is on each individual's estimated class membership, latent class analysis can be thought of as a type of clustering algorithm. In line with this perspective, mixture modeling is sometimes described as "model-based clustering" (Hennig, Meila, Murtagh, & Rocci, 2015; Scrucca, Fop, Murphy, & Raftery, 2017). Many clustering algorithms apply some recursive splitting algorithm to the data. By contrast "model-based" clustering refers to the fact that latent class analysis estimates cluster membership based on a parametric model. Specifically, the posterior class probability that an individual belongs to a latent class can be computed from the likelihood of that individual's observed data under given the class-specific model.

Finally, in the context of machine learning, latent class analysis can be considered as an *unsupervised classification* problem (**figueiredoUnsupervisedLearningFinite2002?**). The term *unsupervised* refers to the fact that the outcome variable, true class membership, is not known, and the term *classification* refers to the fact that the algorithm is predicting a categorical outcome (class membership).

A taxonomy of latent class analyses

In this paper, we use the term latent class analysis to refer to techniques that estimate latent class membership based on a parametric model of observed indicators. From a historical perspective, the term latent class analysis was initially conceived to refer to analyses with categorical (binary) indicators (**vermuntj.k.LatentClassAnalysis2004?**). Nowadays, there are a number of related techniques, known by distinct names, that serve a similar purpose. The term "latent class analysis" seems most appropriate as an umbrella term for this broader class of models, as it only refers to the purpose of the analysis, and does not imply restrictions to the model used, or the level of measurement of the indicators. Given the abundance of terms in use for closely related classes of models, we will provide a rudimentary taxonomy of latent class analyses.

One common type of latent class analyses is the *finite Gaussian mixture model*; a univariate analysis where the observed distribution of a single variable is assumed to result from a mixture of a known number of Gaussian (normal) distributions. The parameters of a finite Gaussian mixture model are the means and variances of these underlying normal distributions. The analysis of shoe sizes presented earlier is a canonical example of this type of analysis. In the multivatiate case, with more than one indicator variable, the parameters of a mixture model are the means, variances, and covariances between the indicators (which can be standardized to obtain correlations). These parameters can be estimated freely, or constrained, across classes.

The technique known as *latent profile analysis (LPA)* is a special case of the mixture model, which assumes conditional independence of the indicators. Conditional independence means that, after class membership is accounted for, the covariances/correlations between indicators are assumed to be zero. This can be conceived of as a restricted mixture model with covariances fixed to zero. In some cases, such constraints will be inappropriate; for example, when the cohesion between indicators is expected to differ between classes. As an example, a mixture model analysis of ocean plastic particles found two classes of particles based on length and width: A class of smaller particles with a high correlation between length and width, meaning that these particles were approximately round or square in shape, and a class of larger particles with a low correlation between length and width, meaning that these particles were heterogenous in shape. From a theoretical perspective, this makes sense, because the smaller particles have been ground down to a more uniform shape by the elements.

It is also possible to estimate a mixture model based on latent indicators. This means that, within each class, one or more continuous latent variables are estimated based on the observed indicators. Categorical latent variable membership is then estimated based on these continuous latent variables. A common application of this approach is in longitudinal research, where the indicators reflect one construct assessed at different time points. Examples of this approach include *growth mixture models* (GMM) and *latent class growth analyses* (LCGA). These techniques estimate a latent growth model to describe individual trajectories over time. The growth mixture model is a latent class model where the parameters that indicate class membership are the intercepts and variances (and typically covariances) of the latent growth variables, e.g., a latent intercept and slope. This technique assumes that individuals within a class can have heterogenous trajectories. If the variance of the growth parameters is fixed to zero, it is known as a latent class growth analysis. This latter approach assumes that all individuals within a class share the same identical trajectory, and that any variance in the indicators not explained by the class-specific latent trajectories is due to residual error variance.

When a latent class analysis relies on categorical indicators, typically of a binary or ordinal measurement level, it has a different parameterization than a mixture model. One common assumption is that each categorical variable reflects an underlying standard normal distribution. The parameters in such a model are then “thresholds”, corresponding to the quantiles of a standard normal distribution (with $N(\mu = 0, \sigma = 1)$). These thresholds are estimated based on the proportion of individuals in each of the response categories of the indicator variable. For example, a binary indicator has a single threshold that distinguishes the two response categories. If responses are distributed 50/50, then the corresponding threshold would be $t_1 = 0.00$. If the responses are distributed 60/40, then the resulting threshold would be $t_1 = 0.25$. This paper will primarily focus on mixture models and special cases thereof, although most of the suggested guidelines are applicable to latent class analyses.

Use cases for latent class analysis

There are several use cases for which latent class analyses are suitable. One example is to test a theory that postulates the existence of a categorical latent variable. For example, *identity status theory* posits that, at any given point in time, adolescents reside in one of four identity statuses. Mixture modeling can be used to identify these four statuses based on observed indicators (e.g., self-reported identity exploration and commitment). If results indicate that the data are better described by a different number of classes, or that the four-class solution does not correspond to the predicted pattern of responses on the indicators, then the theory may be called into question.

Another use case is unsupervised learning; when the goal is to restore unobserved class membership based on observed indicators, or to classify individuals. For example, a mixture model can be used as a diagnostic aid when several clinical indicators can be used to distinguish between a fixed number of physical (**baughmanMixtureModelAnalysis2006?**) or mental (**wuAbuseDependencePrescription2011?**) health problems. The example of shoe size is a rudimentary illustration of this type of application.

Best practices

In estimation

The best practices in estimation, as outlined in Table **??**, are inspired by prior recommendations for best practices for estimating specific sub-types of latent class analyses, including latent class growth analysis (**schootGRoLTSChecklistGuidelinesReporting2017?**) and latent class analysis with ordinal indicators (e.g., **nylund-gibsonTenFrequentlyAsked2018?**). These were generalized to be more relevant to all types of latent class analyses, and updated to current best practices, as explained below.

#	Item
1.	Examine observed data
2.	Handling missing data
3.	Alternative model specifications
4.	Starting values
5.	Algorithm

Examine observed data. Examining observed data is essential for any analysis, as it may reveal patterns and violations of assumptions that had not been considered prior to data collection. Pay special attention to level of measurement of the indicators. Indicators with an ordinal level of measurement are likely to violate the assumption of within-class normal distributions (see **vermuntKmeansMayPerform2011?**). Personal experience consulting on latent class analyses and moderating the `tidyLPA` Google group suggest that the mis-application of mixture models to ordinal (e.g., Likert-type) indicators is perhaps the most common source of user error. Whereas it has been argued that some parametric methods are robust when scales with 7+ indicators are treated as continuous (e.g., **normanLikertScalesLevels2010?**), this certainly does not imply that all methods are. It is certainly unlikely that such ordinal variables can be treated as a *mixture* of multiple normal distributions. The problem becomes eggregious when the number of classes estimated equals or exceeds the number of categories; in this case, each class-specific mean could describe a single response category, and a class-specific variance component would be nonsensical. In sum, Likert-type scales are rarely suitable for mixture modeling; a latent class analysis with ordinal indicators may be more appropriate.

Extensive descriptive statistics (including the number of unique values, variance of categorical variables, and missingness; see next paragraph) can be obtained using the function `tidySEM::descriptives(data)`. Note, however, that sample-level descriptive statistics are of limited value when the goal of a study is to identify sub-samples using latent class analysis. Plots (density plots for continuous variables, and bar charts for categorical ones) may be more diagnostic. Note that density plots can also aid in the choice of the number of classes, as further explained in the section on visualization. Descriptive statistics and plots can be relegated to online supplements, provided that these are readily accessible (consider using a GitHub repository as a comprehensive public research archive, as explained in **vanlissaWORCSWorkflowOpen2020?**).

Missing data. Previous work has emphasized the importance of examining the pattern of missing data and reporting how missingness was handled (**schootGRoLTSChecklistGuidelinesReporting2017?**). Three types of missingness have been distinguished in the literature (**rubinInferenceMissingData1976?**): Missing completely at random (MCAR), which means that missingness is random; missing at random (MAR), which means that missingness is contingent on the *observed* data (and can thus be accounted for); and finally missing not at random (MNAR), which means that missingness is related to unobserved factors. It is possible to conduct a so-called "MCAR" test, for example the non-parametric MCAR test (**jamshidianTestsHomoscedasticityNormality2010?**). But note that the name "MCAR test" is somewhat misleading, as the null-hypothesis of this test is that the data are not MAR, and a significant test statistic indicates that missingness is related to the observed data (MAR). A non-significant test statistic does not distinguish between MCAR or MNAR. As Little's classic MCAR test relies on the comparison of variances across groups with different patterns of missing data, it assumes normality (**littleTestMissingCompletely1988?**). This assumption is tenuous in the context of latent class analysis. A non-parametric MCAR test, as provided by Jamshidian and Jalal, may be more suitable (**jamshidianTestsHomoscedasticityNormality2010?**). Unfortunately, this test was removed from the central R-repository CRAN due to lack of maintenance. For this tutorial, I have re-implemented it in the `mice` package as `mice::MCAR()`, with a fast backend in C++ and new printing and plotting methods.

While we concur that investigating missingness is due dilligence, it is important to emphasize that missingness is adequately handled by default in many software packages for latent class analyses, such as `OpenMx` (and e.g., Mplus). These packages use Full Information Maximum Likelihood (FIML) estimation, which makes use of all available information without imputing missing values. FIML is a best-practice solution for handling missing data; on par with multiple imputation (**leeComparisonFullInformation2021?**). FIML estimation assumes that missingness is either MCAR or MAR. Thus, one would typically proceed with FIML regardless of the outcome of an MCAR test. Although FIML does not, by default, handle missingness in exogenous variables - all indicator variables in latent class analysis are endogenous, so this is not a concern.

Multiple imputation is less suitable to latent class analyses for two reasons. First, because latent class analyses are often computationally expensive, and conducting them on multiple imputed datasets may be unfeasible. Second, because there is no straightforward way to integrate latent class analysis results across multiple datasets. To conclude; our recommendation is to inspect missingness (e.g., using `mice::MCAR()`) and report the number of missings per variable (e.g., using `tidySEM::descriptives()`), before proceeding with FIML. One minor concern is that the K-means algorithm, which `tidySEM` uses for determining starting values, is *not* robust to missing values. When it fails, `tidySEM` automatically switches to hierarchical clustering, unless the user specifies a different clustering algorithm or uses manual starting values.

Alternative model specifications. The different types of latent class models have different parameters. For example, mixture models and latent profile analyses typically have class-specific means, variances, and covariances. Latent growth analyses have the same parameters, but with respect to the latent growth variables. Latent class analyses with ordinal indicators have thresholds. All of these parameters can be freely estimated across classes, or constrained, or fixed (e.g., to zero). The total number of parameters thus scales with the number of estimated classes. Consequently, latent class analyses have a potentially very high number of parameters. As any of these parameters could be mis-specified, it is important to consider alternative model specifications. However, alternative model specifications may be approached differently depending on whether an analysis is data driven (exploratory), or theoretically driven (confirmatory). This distinction has remained underemphasized in prior writing.

Prior literature on latent class analysis has emphasized exploratory applications of the method (see **nylundDecidingNumberClasses2007?**). In exploratory analyses, a large number of models are typically estimated in batch, with varying numbers of classes and model specifications. The "correct" model specification is then determined based on a combination of fit indices, significance tests, and interpretability. For latent profile analysis, the function `tidySEM::mx_profiles(classes, variances, covariances)` largely automates this process. The argument `classes` indicates which class solutions should be estimated (e.g., 1 through 6). The argument `variances` specifies whether variances should be `"equal"` or `"varying"` across classes. The argument `covariances` specifies whether covariances should be constrained to `"zero"`, `"equal"` or `"varying"` across classes. The means are free to vary across classes by default, although the more general function `tidySEM::mx_mixture()` could be used to circumvent this. After all models have been estimated, the function `tidySEM::table_fit()` can be used to obtain a model fit table suitable for determining the optimal model according to best practices. Note however that this table does not include the bootstrapped likelihood ratio test (BLRT) by default, because this test is very computationally expensive. It is recommended to use the function `tidySEM::BLRT()` to compare a shortlist of likely candidate models based on other fit indices. Fit indices typically used for determining the optimal number of classes include the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC). Both information criteria are based on the -2 log likelihood (which is lower for better fitting models), and add a penalty for the number of parameters (thus incentivizing simpler models). This helps balance fit and model complexity. The BIC usually applies a stronger penalty for complexity that scales logarithmically with the sample size.

Fit indices may occasionally contradict each other, so it is important to identify a suitable strategy to reconcile them. One option is to select a specific fit index before analyzing the data. Another option is to always prefer the most parsimoneous model that has best fit according to any of the available fit indices. Yet another option is to incorporate information from multiple fit indices using the analytic hierarchy process (**akogulComparisonInformationCriteria2016?**).

Confirmatory analyses typically require less comprehensive alternative model specifications. For example, in the context of preregistered analyses, the main models of interest may have been specified a priori.

Van de Schoot and colleagues (**vandeschootGRoLTSChecklistGuidelinesReporting2017?**) argue that there are many choices to be made in the specification of latent trajectory models, and that alternative

```
* Consider alternative parameterizations; e.g., free variances/covariances
* See Nylund
```

Software

Many software packages are available for the estimation of latent class analyses. Some of these packages have limited functionality, or implement particular innovations. Other packages implement latent class analyses in the context of a more flexible structural equation modeling framework. The most notable examples of the latter are the commercial programs Mplus and Latent GOLD, and the free open source R-package OpenMx. The commercial packages stand out because they offer relatively user-friendly interfaces and implement sensible defaults for complex analyses, including latent class analysis. This somewhat lowers the threshold for applied researchers to adopt such methods. Commercial software also has several downsides, however. One such downside is that use of the software is restricted to those individuals and institutions who can afford a license. A second downside is that the source code, being proprietary, cannot be audited, debugged, or enhanced by third parties. This incurs the risk that mistakes in the source code may go unnoticed, and curbs progress as software developers cannot add new functionality.

Conversely, the free open source program OpenMx is very flexible, but not very user-friendly.

New functionality in the R-package `tidySEM` seeks to lower the threshold for latent class analysis using `OpenMx`. It adheres to best practices in estimation and reporting, as described in this paper. The user interface is simple, making use of the model syntax of the widely used `lavaan` R-package. This syntax offers a human-readable way to specify latent variable models. Minor enhancements are made to simplify the specification of latent class analysis.

Because of the limitations in ex tools, we set out to develop a tool that a) provided sensible defaults and were easy to use, but provided the option to access and modify all of the inputs to the model (i.e., low barrier, high ceiling), b) interfaced to existing tools, and are able to translate between what existing tools are capable of and what researchers and analysts carrying-out person-oriented analyses would like to specify,

fully-reproducible analyses and

Best practices in estimation

Algorithm. Mixture model parameters and model fit statistics can be estimated in a variety of ways. The choice of the estimator depends on the presence of missing values, sample size, number of indicators, and available computational resources (Weller, Bowen & Faubert, 2020). A commonly used technique is maximum likelihood (ML) estimation with the expectation-maximization (EM) algorithm as a local optimizer. Imagine we are estimating two parameters, e.g. the class-specific means μ_c on a continuous indicator (ignoring the variance for now). The EM algorithm will attempt to find a combination of values for these two parameters that maximizes the likelihood (LL) of all observed data. In practice, instead of maximizing LL, often $-2 * LL$ is minimized, as this offers computational advantages. We can think of this optimization problem as a three-dimensional landscape: The X and Y dimensions are determined by the class-specific means, so $X = \mu_1$ and $Y = \mu_2$ - and the Z-dimension is determined by $Z = -2 * LL$. The optimizer must find the deepest “valley” in this landscape, which reflects the combination of μ_1 and μ_2 that maximizes the likelihood of the data. The EM optimizer behaves somewhat like a marble, dropped in this landscape. It is dropped at some random point in space, and will roll into the nearest valley. The problem is that, once EM rolls into a valley, it will settle on the bottom of that valley (this is known as “convergence”). It cannot climb out again. Thus, if their are multiple valleys, the risk is that the optimizer gets stuck in a shallower valley (a “local optimum”), and never discovers the deepest valley (the “global optimum”, or best solution). One solution to this problem is to drop many marbles at random places, compare their final $-2 * LL$ values, choose the solution with the lowest $-2 * LL$, and make sure that several marbles replicated this solution. This is the “random starts” approach.

One problem with the random starts approach is that it is computationally expensive to run this many replications. Moreover, because the algorithm begins with random starting values, many of the marbles are likely to be very far away from a “good enough” solution. Two innovations may improve the estimation procedure. The first is that, instead of picking random starting values, a “reasonable solution” may be used for the starting values. For example, if we assume that the different classes are likely to have different mean values on the indicators, then the K-means clustering algorithm can be used to determine these cluster centroids. We can compute the expected values of all model parameters by treating the K-means solution as a known class solution, and use these as starting values for a mixture model. One remaining concern is that this approach may result in starting values close to a local optimum, and that the EM algorithm will thus never find the global optimum. A second innovation addresses this concern.

Instead of using EM, it is possible to use an optimizer that can climb out of a valley. Simulated annealing iteratively considers some "destination" in the landscape, and compares its likelihood to the current one. If the destination likelihood is higher, the estimator moves there. If the destination likelihood is *lower*, the estimator still moves there occasionally, based on probability. This latter property allows it to escape local optima.

By default, `tidySEM` employs this solution of deriving starting values using K-means clustering, and identifying the global optimum solution using simulated annealing. Once a solution has been found, simulated annealing is followed up with a short run of the EM algorithm, as EM inherently produces an asymptotic covariance matrix for the parameters that can be used to compute standard errors. Note that these defaults can be manually overridden.

In the case of Latent Profile Analysis, it has been suggested that when tests of multivariate normality indicate a violation, maximum likelihood with robust standard errors should be used (Spurk et al., 2020). This statement is problematic for two reasons. Firstly, multivariate normality will always be violated in mixture models by definition. We assume that the population consists of several groups each characterized by its own normal distribution. When two or more normal distributions with different means and standard deviations comprise a mixed population distribution, this combined distribution is no longer normal. For example, you can imagine a bimodal distribution when dealing with two latent classes. If this were not the case and our mixed population distribution were normal, there would be no latent classes to extract as the population would consists of a single class only. For this reason, multivariate normality will always be violated when the population is comprised of several latent groups. Secondly, the claim that maximum likelihood with robust standard errors should be preferred implicitly implies that only software packages with this estimation technique can be used.

Starting values. The EM algorithm needs starting values for each parameter it wants to estimate. If it were provided a single starting value for each parameter, the EM optimizer would be at risk of finding a local optimum instead of the global optimum. In other words, it would converge on a suboptimal solution resulting in biased parameter and model fit estimates. To prevent this, we should give the algorithm a sufficiently large set of random starting values for each parameter in order to maximize our chances of finding the deepest valley that is the global optimum (McLachlan & Peel, 2004). Hipp and Bauer (2006) recommend using minimally 50-100 sets of random starting values. Also, we should ensure the number of initial stage iterations is large for optimization to be successful (Geiser, 2012).

Furthermore, the choice of the starting values can impact the speed of the algorithm's convergence or in case of having poor initial values the algorithm might diverge altogether (McLachlan & Peel, 2004). The key is to use a large number of starting value sets. Geiser (2013) recommends at least 500 sets of initial values in the first optimization step for simple models, and a greater number of sets for more complex models. A large number of starting values ensures that we find the true maximum and estimate the model parameters and fit accurately.

[(**nylund-gibsonTenFrequentlyAsked2018?**);]

```
In many circumstances and especially in a cluster analysis setting, the mixture componen
```

Assessing Estimation Results. In LCA, a sequence of models is fitted to the data with each additional model estimating one more class than the previous model. The final model called the final class solution is chosen based on both theoretical and statistical criteria. Theory should drive the selection of indicator variables, inform the expectations and reflect on the findings. In addition to this, there are several statistical criteria to consider in model selection. These include but are not limited to likelihood ratio tests, information criteria, and the Bayes factor (Weller, Bowen & Faubert, 2020).

Absolute model fit can be assessed using the likelihood ratio (LR) χ^2 goodness-of-fit test. It can compare two nested models when they have the same number of classes (Lanza et al., 2003). Typically, we compare our final class solution (a constrained model) to an unconstrained one (a model characterized by the perfect fit). The former model is a special, less complex version of the latter. In this case, we compare our model-based response pattern frequencies to those in the dataset. Therefore, we make a statement about how well our model fits the data. The perfect fit would be indicated by the $\chi^2 = 0$ and $p = 1$. Ideally, the LR χ^2 test will be non-significant as this implies that while our constrained model is more parsimonious, it fits the data just as well as the unconstrained model. The underlying assumption of this procedure is that the test statistic is asymptotically distributed as a χ^2 meaning that it can be approximated using this distribution. This is the case when the sample size is adequate (Lanza et al., 2003). The dependence of the LR χ^2 test statistic on sample size has its downsides (Masyn, 2013). Very large samples will often effortlessly reject the null even when the final class solution is a good fit, and small samples will often be underpowered resulting in non-significant χ^2 statistic. In fact, small samples might not be well approximated using the χ^2 distribution, which is a problem shared with large yet sparse datasets. In such cases, special caution is advised when interpreting the resulting p-values (Masyn, 2013).

Relative model fit can be examined using the likelihood ratio test. This is only appropriate when the two models we wish to compare are nested. The likelihood ratio test statistic is computed as the difference in maximum log likelihoods of the two models, with the new degrees of freedom being the difference in their degrees of freedom. This statistic also follows the χ^2 distribution. Similar to the LR χ^2 goodness-of-fit test, we want the test statistic to be non-significant in order to give support to the simpler model. The likelihood ratio test can only compare two models at a time (Lanza et al., 2003).

If we wish to simultaneously compare multiple models based on their relative fit, this can be done through a comparison of multiple information criteria. Examples include the Akaike information criterion (AIC), the Consistent Akaike Information Criterion (CAIC), the Bayesian information criterion (BIC), and the Sample-size Adjusted Bayesian Information Criterion (SABIC). Information criteria are a sum of a measure of fit (usually a form of the converged maximum log likelihood value) and a penalty for model complexity (combination of sample size and number of modeled parameters). As a general rule, the lower the value of an information criterion, the better the model fits the data. When examining several competing models, each with differing number of classes, we expect the information criteria values to drop with each successive model until the final class solution is reached. Further models with additional classes should show worse fit as information criteria reach an optimum before their values rise again with an increasing number of classes. Multiple information criteria should be compared when choosing the final class solution. This can be done by means of an elbow plot (for an example see Nylund-Gibson and Choi, 2018). Advantages of using information criteria are that we can compare multiple models at a time, and these models need not be nested (Masyn, 2013).

When estimators which require sets of starting values are used, each fitted model will have a corresponding log likelihood value. Ideally, the largest log likelihood value of the final class solution will be replicated a large number of times with each log likelihood value generated by a different starting value set. Successful replications of the largest log likelihood value across different starting values instill confidence that the estimation was successful and we indeed found the global maximum (Masyn, 2013). The model with the highest replicated likelihood will be chosen as our best-fit. If the largest log likelihood value is not replicated many times, the model may be unidentified (Geiser, 2012). Such a model does not have a unique solution. For a discussion on model identification for LCA, see Masyn (2013). Nylund-Gibson and Choi (2018) recommend that the highest log likelihood should be replicated in at least 3-10% of the models initialized with different starting values in the first step of the optimization. This consideration does not apply in case of models which use simulated annealing since these do not have starting values.

Classification Diagnostics. Best models will divide the sample into subgroups which are internally homogeneous and externally distinct. Classification diagnostics give us a way to assess the degree to which this is the case. They are separate from the absolute and relative goodness-of-fit as a model can fit the data well but show poor latent class separation (Masyn, 2013). A fundamental concept when examining classification precision and accuracy are the posterior class probabilities. A probability of belonging to each latent class is computed for each individual. The highest posterior class probability is then determined and the individual is assigned to the corresponding class. We want each individual's posterior class probabilities to be high for one and low for the remaining latent classes. This is considered a high classification accuracy and means that the classes are distinct. Three important classification diagnostic measures are entropy, the average posterior probability and the modal class assignment proportion.

Entropy is a summary measure of posterior class probabilities across classes and individuals. It ranges from 0 (model classification no better than random chance) to 1 (perfect classification). As a rule of thumb, values above .80 are deemed acceptable and those approaching 1 are considered ideal. Entropy should not be used to select a particular class solution. An appropriate use of entropy is that it can disqualify certain solutions if class separability is too low or if one of the latent classes is too small to be meaningful or to calculate descriptive statistics.

The average posterior probability is a measure of classification uncertainty for each latent class.

The modal class assignment proportion

Conditional item probabilities indicate the probability of a positive score on an item given that the person belongs to a particular latent class. However, if many conditional item probabilities in estimation are found to be extreme (as in exactly 0 or 1), we are observing boundary parameter estimates. This can be a sign of an invalid solution, warn us that too many classes were extracted, or indicate a local optimum. Boundary parameter estimates are more likely to happen when working with sparse data and should be interpreted with caution (Geiser, 2012).

An important outcome of LCA are unconditional class probabilities. These communicate the probability that an individual belongs to each class. A limitation is that our final class solution need not reflect true class membership. Once we do accept a particular class solution, special care also needs to be taken when naming the classes. Class names should be chosen to accurately reflect group membership. Overly simplified and generalized class names may prove misleading to both audiences and researches alike leading to what is known as a naming fallacy (Weller, Bowen & Faubert, 2020).

Label switching. The final class solution will usually discover and enumerate several classes. The class ordering however is completely arbitrary. The class labeled as Class 1 in one solution may become Class 2 or Class 3 in another model, even when the only difference between the models is in their starting values. Label switching is something to be mindful of when comparing different LCA models (Masyn, 2013).

Best practices in reporting

Among studies using LCA, reporting practices vary significantly (Weller, Bowen & Faubert, 2020). Various authors have tried to better and standardize ways of reporting LCA (e.g. Masyn, 2013; Weller, Bowen & Faubert, 2020). Building on their work, we provide guidelines with a strong emphasis on scientific reproducibility and transparency better known as the open science framework (OSF). Open science framework promotes scientific openness, reproducibility and integrity through an establishment of guidelines for the scientific process (Foster & Deardorff, 2017) To this end, van Lissa and colleagues

(2020) developed WORCS, a workflow for open reproducible code in science. WORCS consists of step-by-step guidelines for research projects based on the TOP-guidelines developed by Nosek and colleagues (2015). It can be easily implemented in R in form of an R package which facilitates preregistration, article drafting, version control, citation and formatting, among others (Van Lissa et al., 2020)

```
* Use comprehensive citation; this includes referencing the software used.
* Share code and (if possible) data. Van Lissa and colleagues suggest sharing synthetic
* Ideally, make the entire research project reproducible so that others may download it
```

As the open science movement is gaining momentum, researchers are becoming increasingly aware how important it is that analyses can be reproduced and audited. In line with open science principles, one of the suggested reporting standards relates to reproducible code. In this context, it is important to note that user-friendly methods for estimating latent class analyses have predominantly been available in commercial software packages (e.g., *Mplus* and *Latent GOLD*). A potential downside of commercial software is that it restricts the ability to reproduce analyses to license holders, and prevents auditing research because the underlying source code is proprietary. To overcome these limitations, the present paper introduces new user-friendly functions in the `tidySEM` R-package that can be used to estimate a wide range of latent class analysis models using the free, open-source R-package `OpenMx`. The reporting guidelines described in this paper are adopted in `tidySEM` by default. The `tidySEM` R-package thus makes advanced mixture modeling based on best practices widely accessible, and facilitates the adoption of the estimation and reporting guidelines described in this paper.

Best practices in visualization

Tutorial

–> –> –>

–>

–> –> –>

–> –> –> –> –> –> –> –>

–> –>

–> –> –> –> –> –> –> –> –> –> –> –> –> –> –>

–> –>

–> –> –>

–> –> –> –> –> –> –> –> –> –> –>

–> –> –> –> –> –> –> –> –> –> –> –> –> –> –>

–> –> –> –> –> –> –> –> –> –> –> –> –> –> –> –> –>

–> –> –> –> –> –> –> –> –> –> –> –> –> –> –> –> –> –> –> –>

Starting values.

Table 1

Observed group membership by estimated class membership.

Observed	Class 1	Class 2
Man	21	28
Woman	51	0

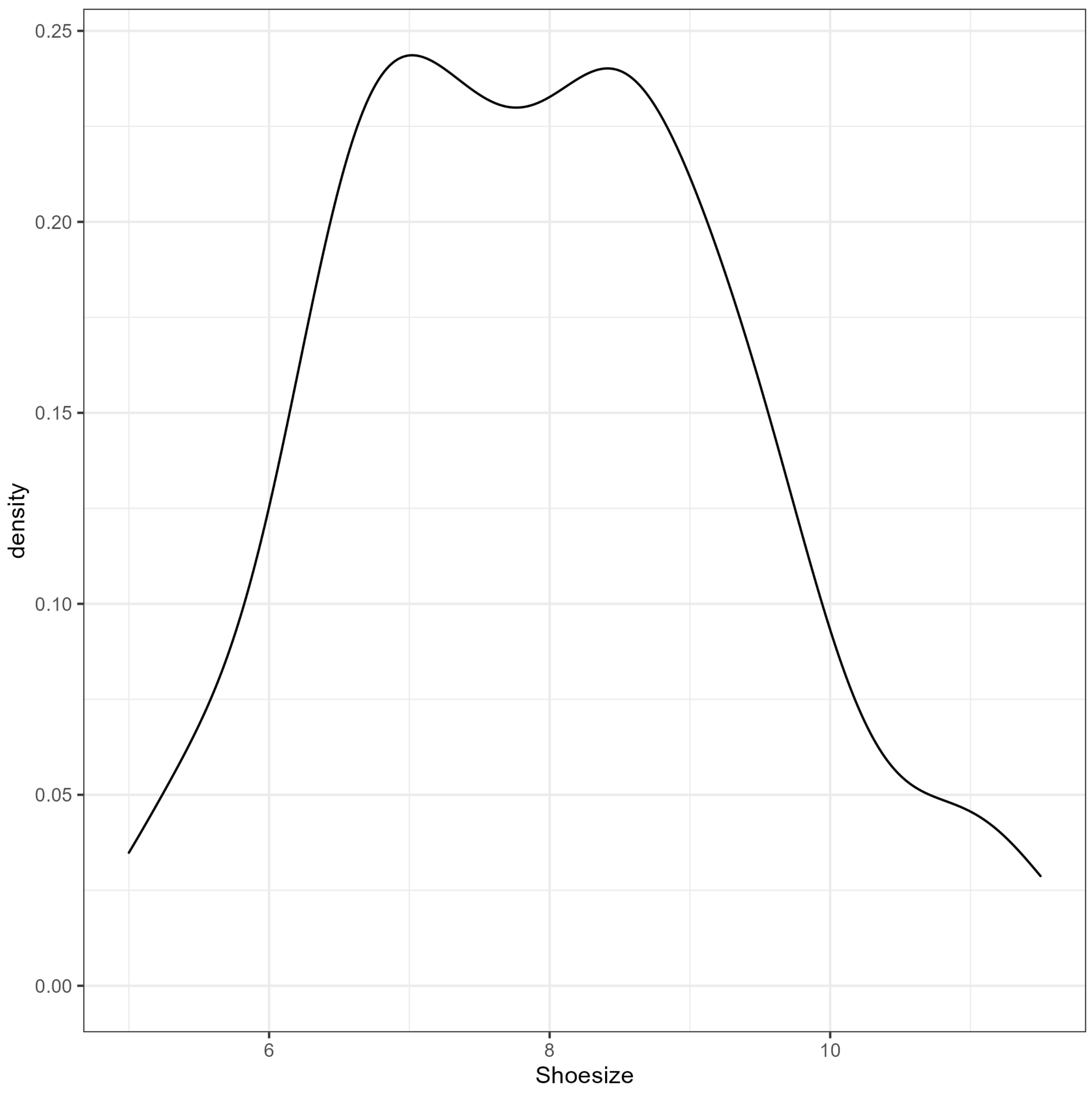

Figure 1. Kernel density plot of shoe sizes.