How can I estimate a multiple group latent class model with known classes?

Authored by

stats writer

July 1, 2024

A multiple group latent class model is a statistical method used to analyze data that involves multiple groups with known classes. This model allows for the identification of underlying latent classes within each group and the estimation of the relationships between these classes. To estimate a multiple group latent class model, one must first define the known classes within each group and then use statistical software to estimate the model parameters. This process involves specifying the number of latent classes, determining the best fitting model based on various statistical measures, and interpreting the results to understand the relationships between the identified latent classes. This approach can provide valuable insight into the underlying structure of the data and aid in making informed decisions in various fields such as marketing, psychology, and social sciences.

How can I estimate a multiple group latent class model (knownclass)? | Mplus FAQ

This page was created using Mplus version 5.2, the output and/or syntax may be different for other versions of Mplus.

Frequently, we wish to compare the structure of measurement models across groups (e.g. men and women). When the latent variable is categorical the model is often referred to as a latent class analysis (LCA), more generally, these models are sometimes referred to as mixture models. Below we show how to estimate an LCA with either continuous or categorical class indicators (it is also

possible to estimate a
model with both categorical and continuous class indicators). We will start with
a latent class model with continuous indicators, because these models have a
slightly
simpler syntax. In Mplus, the knownclass option is used to estimate a
latent class model with multiple groups. This option takes
its name from the fact that the grouping variable (e.g. gender) is known (i.e.
observed).

In the examples below, group is the known or observed class, while
c is the latent variable estimated using the observed items. There
are three continuous observed items, named a1, a2, and a3. You can
download the example dataset here:
https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_con.dat .

A single group latent class model

As a starting place, below we show the syntax for a single group latent class model.
In this model, the continuous variables a1, a2, and a3, are
used to form a latent variable c with two classes. The file option of the data: command gives the name of the file in which the dataset is stored. In the variable: command the
names option gives the names of the variables in the dataset. The
usevariables option gives the names of the variables used to estimate the
model. The classes
option defines the names of the categorical latent variable c,
followed by the number of classes in parentheses, that is (2) for a two
class latent variable. In the analysis: command, the type = mixture
command indicates that we wish to estimate a mixture model.

data:

file = https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_con.dat ;
variable:
names = group a1 a2 a3;
usevariables = a1 a2 a3;
classes = c(2);
analysis:
type = mixture;

Model allowing differences in item means across groups, fixing class probabilities and item variances across groups and classes

In this model, we add the observed grouping variable, group to our model in order to estimate a multiple group mixture model. In this model, the classes option of the variable: command lists two classes (c and g), each with the number of groups listed in parentheses after the class name. The knownclass option specifies that the classes in the variable g are defined by the

observed variable group, the observed values of group associated with each
class (e.g. group = 0) are listed in parentheses after the class name (i.e.
g).

```
data:
file = https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_con.dat;
variable:
names = group a1 a2 a3;
usevariables = a1 a2 a3;
classes = c(2) g(2);
knownclass = g (group=0 group=1);
analysis:
type = mixture;
```

Model allowing differences in item means and class probabilities across groups, with item variances fixed across groups and classes

In this model we use g (i.e. the grouping variable) to predict the
probability of class membership in c, meaning that the

probability of
being in a given class is allowed to vary by the observed variable group. First, we have changed the classes option so that
the known class (i.e. g) is listed first, this is necessary if we want to
regress c on g to allow the class probabilities to vary by level
of group. We have also added the model: command, in the overall
section of the model (under %overall%), we have added c on g which
adds a regression in which the known class variable (g) predicts the
probability that a given case will be in one of the classes of the latent
variable c, this allows the class probabilities for c to vary by
g.

data:
file = https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_con.dat ;

variable:
names = group a1 a2 a3;
usevariables = a1 a2 a3;
classes = g(2) c(2);
knownclass = g (group=0 group=1);
analysis:
type = mixture;
model:
%overall%
c on g;

Another model allowing differences in item means and class probabilities across groups, fixing item variances across groups and classes

The model estimated in this example is identical to the previous model, using
different syntax. In the model below we have explicitly listed the item means in the input
file, so that we can fix or free individual parameters across groups, which
allows us to test for differences between item means in the two groups. We can
confirm that the two models are the same by comparing their log likelihoods, if

the two are the same, we have indeed run the same model. One thing to note is
that when we change to this syntax, the order of the classes may change. This
change is substantively and mathematically unimportant, we are, after all, still running the same model, but it can be hard to keep track of which classes correspond across the
groups if they are not in the same order. To avoid confusion, we can use the
coefficient estimates from the above model as starting values for the current
model. We could use all of the item means, but it turns out this is unnecessary,
a few starting values is usually sufficient to put the classes in the desired
order. Below is the Mplus output from the model immediately above this one (i.e.
the model that is identical, but has different syntax). Here we show the item
means for all three variables, for g = 0 and c = 1 (Latent Class
Pattern 1 1), followed by the item means for g=1 and c=1 (Latent

Class Pattern 2 1).

Two-Tailed
Estimate S.E. Est./S.E. P-Value

Latent Class Pattern 1 1

Means
A1 0.247 0.058 4.227 0.000
A2 2.142 0.091 23.617 0.000
A3 -0.960 0.067 -14.223 0.000

<output omitted>

Latent Class Pattern 2 1

Means
A1 0.978 0.064 15.260 0.000
A2 0.173 0.042 4.120 0.000
A3 -1.541 0.065 -23.570 0.000

The item means for class 1 (of the latent variable c) in each group (g)
are shown above. In the syntax below, we use these parameter estimates as starting values.
Most of the syntax shown below is the same as that

from the previous models, the
new syntax all appears at the end of the input. Below the overall portion of the
model (%overall%) we see specific commands for
each combination of g and c, in this case, both g and c have two categories, so there are four sections of the model. The first section
is for g=1 (group=0) and c=1 is indicated by %g#1.c#1%, the observed (knownclass) variable must come first, if we used
%c#1.g#1% Mplus would issue an error message and the model would not run.
Below this designation we see the syntax describing the structure of the model , this specifically lists each of the item means for the variables
used to form the latent variable c (i.e. a1, a2, and a3). In addition to listing the parameters,
we assign each parameter a starting value based on the above output, for
example, a1*0.247 sets the starting value of the mean of a1 to
0.247 in the class c=1, for the group g=1. For c=2 in g=1 (under the label %g#1.c#2%), we specify the means of the items

a1-a3, but do not assign starting
values, the set of starting values above is sufficient to set the class ordering
for g=1. For c=1 in g=2 (labeled %g#2.c#1%), we
again include starting values, so that the classes for the second group (g=2)
will be in the desired order.

data:
file = https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_con.dat;
variable:
names = group a1 a2 a3;
usevariables = a1 a2 a3;
classes = g(2) c(2);
knownclass = g (group=0 group=1);
analysis:
type = mixture;
model:
%overall%
c on g;

%g#1.c#1%

;

%g#1.c#2%

;

%g#2.c#1%

;

%g#2.c#2%

;

Testing for differences in parameter estimates

Once we have estimated a model in which the item means are allowed to vary
across groups, we may want to test to see whether the differences in item means
between the two groups are significant, one method of doing this is to use the
model test: command to perform a Wald test. Below we test whether the mean
for a1 is different in class=1 across the two groups. To do this we have
given each of the parameters in question a name. In Mplus, parameter names must

appear at the end of a line and in parentheses, for example (p1)
used below gives the mean of a1 for class 1 (c=1) in group 1 (g=1)
the name p1. We have also given the parameter for the mean of a1
in class 1 (c=1) in group 2 (g=2) a name, p2. Note that these names are arbitrary, except that they must begin with a letter and be
enclosed in parentheses. Finally, at the bottom of the input, we use the
model test: command, below this is the test we want to perform,
specifically, we want to test whether p1 = p2.

data:
file = https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_con.dat;
variable:
names = group a1 a2 a3;
usevariables = a1 a2 a3;
classes = g(2) c(2);
knownclass = g (group=0 group=1);

analysis:
type = mixture;
model:
%overall%
c on g;

%g#1.c#1%
(p1);
;

%g#1.c#2%
;

%g#2.c#1%
(p2);
;

%g#2.c#2%
;

model test:
p1 = p2;

The output from this model will have an additional section, shown below. In

this case we can reject the null hypothesis that p1 = p2. If we wanted to
constrain these two parameters to equality (either because the difference was
non-significant, or for other reasons) we could do so by either giving the two
parameters the same name, or replacing the parameter name with a number that is
the same for all parameters that are to be constrained to equality, for example,
we could replace both (p1) and (p2) above with (1).

Wald Test of Parameter Constraints

Value 71.900
Degrees of Freedom 1
P-Value 0.0000

Model allowing differences in item variances across groups, item means and
class probabilities fixed across groups

In this model, we have modified the model: command so that the item
means and class probabilities are fixed across groups

(g), but the item
variances are allowed to vary by group. Under
model c: we describe the structure of the latent variable c.
Under %c#1% the model for class 1 of the latent variable c is defined by the mean of the
variables a1, a2, and a3, this is indicated by the name of the variables listed within square brackets (i.e.). Under %c#2% we describe the
structure of c=2 in the same manner as c=1. Under model g: we explicitly
show the variance structure of the model. Under %g#1% we see the names of the
variables that make up the latent variable c (i.e. a1, a2, and a3),
the name of a variable listed without other commands (e.g. on, with,
or square brackets) indicates the variance of the variable. So listing the
variances of the variables separately by level of g indicates that the
variances of a1, a2, and a3 should be allowed to vary by group.
This syntax is repeated for the second group (g=2),

allowing the
variances for this group to differ from those in the first.

data:
file is
https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_con.dat ;
variable:
names are group a1 a2 a3;
classes = g(2) c(2);
knownclass = g (group=0 group=1);
analysis:
type = mixture;
model:
model c:
%c#1%
;
%c#2%
;
model g:
%g#1%
a1 a2 a3;
%g#2%
a1 a2 a3;

Working with categorical observed variables

In the examples below, instead of the observed variables being continuous, as above, the observed items are categorical. As above, the variable group is the known or observed class variable. The latent variable c is estimated using the categorical observed items named i1, i2, and i3. The variables i1 and i2 are dichotomous, while the variable i3 is ordinal with three categories. You can download the example dataset here: https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_cat.dat.

A single group latent class model

As a starting place, below we show the syntax for a single group latent class model. In this model, the categorical variables i1, i2, and i3, are used to form a latent variable c with two classes. Most of this input file is the same as the single group latent class model

with continuous
indicators. The file option of the data: command gives the name of the
file in which the dataset is stored. In the variable: command the
names option gives the names of the variables in the dataset. The
usevariables option gives the names of the variables used to estimate the
model because not all variables in the dataset are used in the model. The classes
option defines the names of the categorical latent variable c,
followed by the number of classes in parentheses, that is (2) for a two
class latent variable. In the analysis: command, the type = mixture
command indicates that we wish to estimate a mixture model. The difference
between this model, and a single group model with continuous indicators is that
in the variable: command, the categorical option lists the names of the
categorical variables in the dataset, in this case (i.e. i1,

i2, and
i3).

data:
file =
https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_cat.dat ;
variable:
names = group i1 i2 i3;
usevariables = i1 i2 i3;
classes = c(2);
categorical = i1 i2 i3;
analysis:
type = mixture;

Model allowing differences in item thresholds across groups, fixing class probabilities across groups and classes

In this model, we add the observed grouping variable, group to our
model in order to estimate a multiple
group mixture model. In this model, the classes option of the
variable: command lists two classes (c and g), each with the

number of groups listed in parentheses after the class name. The knownclass
option specifies that the classes in the variable g are defined by the
observed variable group, the observed values of group associated with each
class (e.g. group = 0) are listed in parentheses after the class name (i.e.
g).

```
data:
file is https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_cat.dat;
variable:
names = group i1 i2 i3;
usevariables = i1 i2 i3;
classes = c(2) g(2);
knownclass = g (group=0 group=1);
categorical = i1 i2 i3;
analysis:
type = mixture;
```

Model allowing differences in item thresholds and class probabilities across groups

In this model we use g (i.e. the grouping variable) to predict the probability of class membership in c, meaning that the probability of being in a given class is allowed to vary by the observed variable group. First, we have changed the classes option so that the known class (i.e. g) is listed first, this is necessary if we want to regress c on g to allow the class probabilities to vary by level of group. We have also added the model: command, in the overall section of the model (under %overall%), we have included c on g which adds a regression in which the known class variable (g) predicts the probability that a given class will be in one of the classes of the latent variable c, this allows the class probabilities for c to vary by

g.

data:
file is https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_cat.dat ;
variable:
names are group i1 i2 i3;
usevariables are i1 i2 i3;
classes = g(2) c(2);
knownclass = g (group=0 group=1);
categorical = i1 i2 i3;
analysis:
type = mixture;
model:
%overall%
c on g;

Another model allowing differences in item thresholds and class probabilities across groups

The model estimated in this example is identical to the previous model, but uses different input. In the input below we have explicitly listed the item thresholds in the input

file, so that we can fix or free individual parameters across groups, which
allows us to test for differences between item means in the two groups. We can
confirm that the two models are the same by comparing their log likelihoods,
which will match if we have indeed run the same model. One thing to note is
that when we change to this syntax, the order of the classes may change. This
change is substantively unimportant, we are, after all, still running the same
model, but it can be hard to keep track of which classes correspond across the
groups if they are not in the same order. To avoid confusion, we can use the
coefficient estimates from the above model as starting values for the current
model. We could use all of the item thresholds, but a few starting values is usually sufficient to put the classes in the desired
order. Below is the Mplus output from the model immediately above this one (i.e.
the model that is identical, but has different syntax).

Here we show the item
thresholds for all three variables, for g = 0 and c = 1 (Latent Class
Pattern 1 1), followed by the item thresholds for g=1 and c=1
(Latent Class Pattern 2 1), note that there are four thresholds, one each for
i1 and i2 (which have two categories), and two thresholds for i3,
which has three ordinal categories. Each threshold is denoted by the variable
name, followed by a dollar sign and a number indicating the order of the
threshold, for example, i3$1 is the first threshold for i3, and i3$2 is
the second.

Two-Tailed
Estimate S.E. Est./S.E. P-Value

Latent Class Pattern 1 1

Thresholds
I1$1 -0.307 0.689 -0.446 0.656
I2$1 1.976 3.089 0.640 0.522

I3$1 -1.467 1.150 -1.276 0.202
I3$2 1.097 0.304 3.604 0.000

<output omitted>

Latent Class Pattern 2 1

Thresholds
I1$1 1.501 1.453 1.033 0.301
I2$1 -1.870 3.996 -0.468 0.640
I3$1 -0.139 0.294 -0.474 0.635
I3$2 4.358 9.937 0.439 0.661

The item thresholds for class 1 (of the latent variable c) in each group (g)
are shown above. In the syntax below, we use these parameter estimates as starting values.
Most of the syntax shown below is the same as that from the previous models, the
new syntax all appears at the end of the input. Below the overall portion of the
model (%overall%) we see the portions of the model: command for
each combination of g and c, in this case, both g and c have two categories, so there are four sections of the

model. The first section
is for g=1 (group=0) and c=1 is indicated by %g#1.c#1%, the observed (knownclass) variable must come first, if we used
%c#1.g#1% Mplus would issue an error message and the model would not run.
Below this designation we see the syntax describing the structure of the model
, this specifically lists each of the item means for the variables
used to form the latent variable c (i.e. i1, i2, and i3). In addition to listing the parameters,
we assign each parameter a starting value based on the above output, for
example, i1$1*-0.307 sets the starting value of the threshold of i1 to
-0.307 in the class c=1, for the group g=1. For c=2 in g=1 (under the label %g#1.c#2%), we specify the thresholds of the items
i1-i3, but do not assign starting
values, the set of starting values above is sufficient to set the class ordering
for g=1. For c=1 in g=2 (labeled %g#2.c#1%), we
again include starting values, so that the classes for the

second group (g=2)
will be in the desired order.

```
data:
file = https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_cat.dat ;
variable:
names = group i1 i2 i3;
usevariables = i1 i2 i3;
classes = g(2) c(2);
knownclass = g (group=0 group=1);
categorical = i1 i2 i3;
analysis:
type = mixture;
model:
%overall%
c on g;

%g#1.c#1%
;

%g#1.c#2%
;
```

%g#2.c#1%

;

%g#2.c#2%

;

Testing for differences in parameter estimates

Once we have estimated a model in which the item thresholds are allowed to vary
across groups, we may want to test to see whether the differences in item
thresholds
between the two groups are significant. One method of doing so is to use the
model test: command to perform a Wald test. Below we test whether the
threshold 1
for i1 is different in class=1 across the two groups. To do this we have
given each of the parameters in question a name. In Mplus, parameter names must
appear at the end of a line and in parentheses, for example
(p1) used below gives the threshold for i1 in class 1

(c=1) in group 1 (g=1)
the name p1. We have also given the parameter for the threshold of i1
in class 1 (c=1) and group 2 (g=2) a name, p2. Note that these names are arbitrary, except that they must begin with a letter and be
enclosed in parentheses. Finally, at the bottom of the input, we see the
model test: command, below this is the test we want to perform,
specifically, we want to test whether p1 = p2.

```
data:
file = https://stats.idre.ucla.edu/wp-content/uploads/2016/02/mult_grp_lca_cat.dat ;
variable:
names = group i1 i2 i3;
usevariables = i1 i2 i3;
classes = g(2) c(2);
knownclass = g (group=0 group=1);
categorical = i1 i2 i3;
analysis:
type = mixture;
```

model:
%overall%
c on g;

%g#1.c#1%
(p1);
;

%g#1.c#2%
;

%g#2.c#1%
(p2);
;

%g#2.c#2%
;

model test:
p1 = p2;

The output from this model will have an additional section, shown below. In
this case we fail to reject the null hypothesis that p1 = p2. If we wanted to

**constrain these two parameters to equality (either because the difference was
non-significant, or for other reasons) we could do so by either giving the two
parameters the same name, or replacing the parameter name with a number that is
the same for all parameters that are to be constrained to equality, for example,
we could replace both (p1) and (p2) above with (1).**

Wald Test of Parameter Constraints

**Value 1.266
Degrees of Freedom 1
P-Value 0.2605**