Appendix - Implement LCA Analyses with MplusAutomation

_____

Motivation for MplusAutomation workflow:

This `R` tutorial is intended to provide a template for estimating latent class analysis (LCA), integrating the unconditional LCA with a larger SEM model, and producing publication ready figures in a systematic manner. Using this approach all code for data pre-processing, model estimation, tabulation, and figure processing can be contained within a single script providing clear documentation. It is the authors' belief that this dramatically reduces risk of user-error, is conducive to open-science philosophy, and scientific transparency. All models are estimated using `Mplus` (Muthén & Muthén, 1998 - 2017) using the wrapping package `MplusAutomation` (Hallquist & Wiley, 2018). This method requires the user to have the proprietary software `Mplus` installed on their computer.

This approach also relies on the utility of `R-Projects`. This provides a structured framework for organizing all associated data files, Mplus input/output files, scripts, and figures. Given the large number of Mplus output files inherent to fitting mixture models, creating a system of project sub-folders greatly improves organization (i.e., folders; 'data', 'mplus_files' 'figures', etc.) Additionally, the communication between `R` and `Mplus` requires the specification of file-paths, a procedure which is streamlined by use of `R-projects`. Due to the reliance on file-paths, the `here` package ensures that the syntax for file-paths is uniform across operating systems (e.g., Windows or Mac) enabling reproducibility.

_____

Preparation

Download the R-Project

Link to Github repository here: https://github.com/garberadamc/LCA-COPING-BD

For readers **unfamiliar with Github** and version controlled R-projects:

1. On the repository page, click the green `Code` button and in the menu choose option `Download ZIP`
2. Place the un-zipped downloaded folder on your desktop
3. Within this folder open the file with the blue cube icon that is file named `LCA-COPING-BD.Rproj`
4. Next open the file containing all analysis code named `rcode-lca-coping-bd.R`.

Note: If preferred users may follow analyses using the Rmarkdown script (`.Rmd`).

Project folder organization

The following sub-folders will be used to contain files:

1. "data"; 2. "enum_mplus"; 3. "3step_mplus"; 4. "figures"

Note regarding project location: If the main project folder is located within too many nested folders it may result in a file-path error when estimating models with `MplusAutomation`.

_____

To install package {`rhdf5`}

```
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")

BiocManager::install("rhdf5")
```

Load packages

```
library(MplusAutomation)
library(tidyverse)
library(rhdf5)
library(here)
library(glue)
library(gt)
library(reshape2)
library(cowplot)
library(patchwork)
```

Read in data. File-paths are set using R-Projects & the {`here`} package

```
ies_data <- read_csv(here("data","ies_bd_data.csv"),
                     na=c("",".", "999"))
```

_____

1.1 Enumeration:

Estimate K-class models with 1 through 6 classes.

_____

Specification details that are specific to applied coping example:

- Within the `lapply()` function `1:6` indicates the number of K-class models to estimate.
- The measurement model item names for the coping strategy example are listed after the `categorical = ...;` & `usevar = ...;` statements.
- Note, that for Latent Profile Analysis (LPA) the `categorical = ...;` statement would be removed so that items are estimated as continuous variables.

```r
lca_k1_6  <- lapply(1:6, function(k) {
  lca_enum  <- mplusObject(

  TITLE = glue("Class-{k} LCA Enumeration - Youth Coping Strategies"),

  VARIABLE = glue(
    "categorical = do1 do2 do3 do5 do6; !!! Coping strategy items for measurement model !!!
     usevar = do1 do2 do3 do5 do6;

     classes = c({k});"),

  ANALYSIS =
    "estimator = mlr;
     type = mixture;
     processors=10;",

  PLOT =
    "type = plot3;
     series = do1 do2 do3 do5 do6(*);",

  OUTPUT = "sampstat tech11 tech14;",

  usevariables = colnames(ies_data),
  rdata = ies_data)

lca_enum_fit <- mplusModeler(lca_enum,
                  dataout=glue(here("enum_mplus", "c_lca_enum_bd.dat")),
                  modelout=glue(here("enum_mplus", "c{k}_lca_enum_bd.inp")) ,
                  check=TRUE, run = TRUE, hashfilename = FALSE)
})
```

1.2 Generate Model Fit Summary Table

- This syntax can be used to compare model fit from the series of LCA models generated during enumeration (*Table 4* in manuscript).
- The code produces a table that is approximately in APA format.

```r
output_enum <- readModels(here("enum_mplus"), quiet = TRUE)

enum_extract <- LatexSummaryTable(output_enum,
               keepCols=c("Title", "Parameters", "LL", "BIC", "aBIC",
                          "BLRT_PValue", "T11_VLMR_PValue","Observations"))
```

Calculate relevant fit indices for summary table

```
allFit <- enum_extract %>%
  mutate(aBIC = -2*LL+Parameters*log((Observations+2)/24)) %>%
  mutate(CIAC = -2*LL+Parameters*(log(Observations)+1)) %>%
  mutate(AWE = -2*LL+2*Parameters*(log(Observations)+1.5)) %>%
  mutate(SIC = -.5*BIC) %>%
  mutate(expSIC = exp(SIC - max(SIC))) %>%
  mutate(BF = exp(SIC-lead(SIC))) %>%
  mutate(cmPk = expSIC/sum(expSIC)) %>%
  select(1:5,9:10,6:7,13,14) %>%
  arrange(Parameters)
```

Create table using {gt}

```
allFit %>%
  mutate(Title = str_remove(Title, " LCA Enumeration - Youth Coping Strategies")) %>%
  gt() %>%
  tab_header(
    title = md("**Model Fit Summary Table**"), subtitle = md(" ")) %>%
  cols_label(
    Title = "Classes",
    Parameters = md("Par"),
    LL = md("*LL*"),
    T11_VLMR_PValue = "VLMR",
    BLRT_PValue = "BLRT",
    BF = md("BF"),
    cmPk = md("*cmP_k*")) %>%
  tab_footnote(
    footnote = md(
    "*Note.* Par = parameters; *LL* = log likelihood;
      BIC = bayesian information criterion;
      aBIC = sample size adjusted BIC; CAIC = consistent Akaike information criterion;
      AWE = approximate weight of evidence criterion;
      BLRT = bootstrapped likelihood ratio test p-value;
      VLMR = Vuong-Lo-Mendell-Rubin adjusted likelihood ratio test p-value;
      cmPk = approximate correct model probability."),
    locations = cells_title()) %>%
  tab_options(column_labels.font.weight = "bold") %>%
  fmt_number(10,decimals = 2,
             drop_trailing_zeros=TRUE,
             suffixing = TRUE) %>%
  fmt_number(c(3:9,11), decimals = 0) %>%
  fmt_missing(1:11, missing_text = "--") %>%
  fmt(c(8:9,11), fns = function(x)
    ifelse(x<0.001, "<.001",
           scales::number(x, accuracy = 0.01))) %>%
  fmt(10, fns = function(x)
    ifelse(x>100, ">100",
           scales::number(x, accuracy = .1)))
```

<div style="text-align:center">

Model Fit Summary Table[1]

</div>

Classes	Par	LL	BIC	aBIC	CIAC	AWE	BLRT	VLMR	BF	cmP_k
Class-1	5	$-1,294$	$2,619$	$2,603$	$2,624$	$2,664$	–	–	>100	<.001
Class-2	11	$-1,259$	$2,586$	$2,551$	$2,597$	$2,686$	<.001	<.001	>100	1.00
Class-3	17	$-1,249$	$2,602$	$2,548$	$2,619$	$2,757$	<.001	0.12	>100	<.001
Class-4	23	$-1,244$	$2,629$	$2,556$	$2,652$	$2,838$	0.27	0.01	>100	<.001
Class-5	29	$-1,242$	$2,662$	$2,570$	$2,691$	$2,926$	0.67	0.46	>100	<.001
Class-6	35	$-1,241$	$2,695$	$2,584$	$2,730$	$3,014$	0.67	0.49	–	<.001

[1]Note. Par = parameters; LL = log likelihood; BIC = bayesian information criterion; aBIC = sample size adjusted BIC; CAIC = consistent Akaike information criterion; AWE = approximate weight of evidence criterion; BLRT = bootstrapped likelihood ratio test p-value; VLMR = Vuong-Lo-Mendell-Rubin adjusted likelihood ratio test p-value; cmPk = approximate correct model probability.

1.3 Plot Information Criteria

```
allFit %>% select(2:7) %>%
  rowid_to_column() %>%
  pivot_longer(`BIC`:`AWE`,
    names_to = "Index",
    values_to = "ic_value") %>%
  mutate(Index = factor(Index,
    levels = c("AWE","CIAC","BIC","aBIC"))) %>%
  ggplot(aes(x = rowid, y = ic_value,
    color = Index, shape = Index,
    group = Index, lty = Index)) +
  geom_point(size = 2.0) + geom_line(size = .8) +
  scale_x_continuous(breaks = 1:6) +
  scale_colour_grey(end = .5) +
  theme_cowplot() +
  labs(x = "Number of Classes", y = "Information Criteria Value") +
  theme(legend.title = element_blank(),
        legend.position = "top")
```
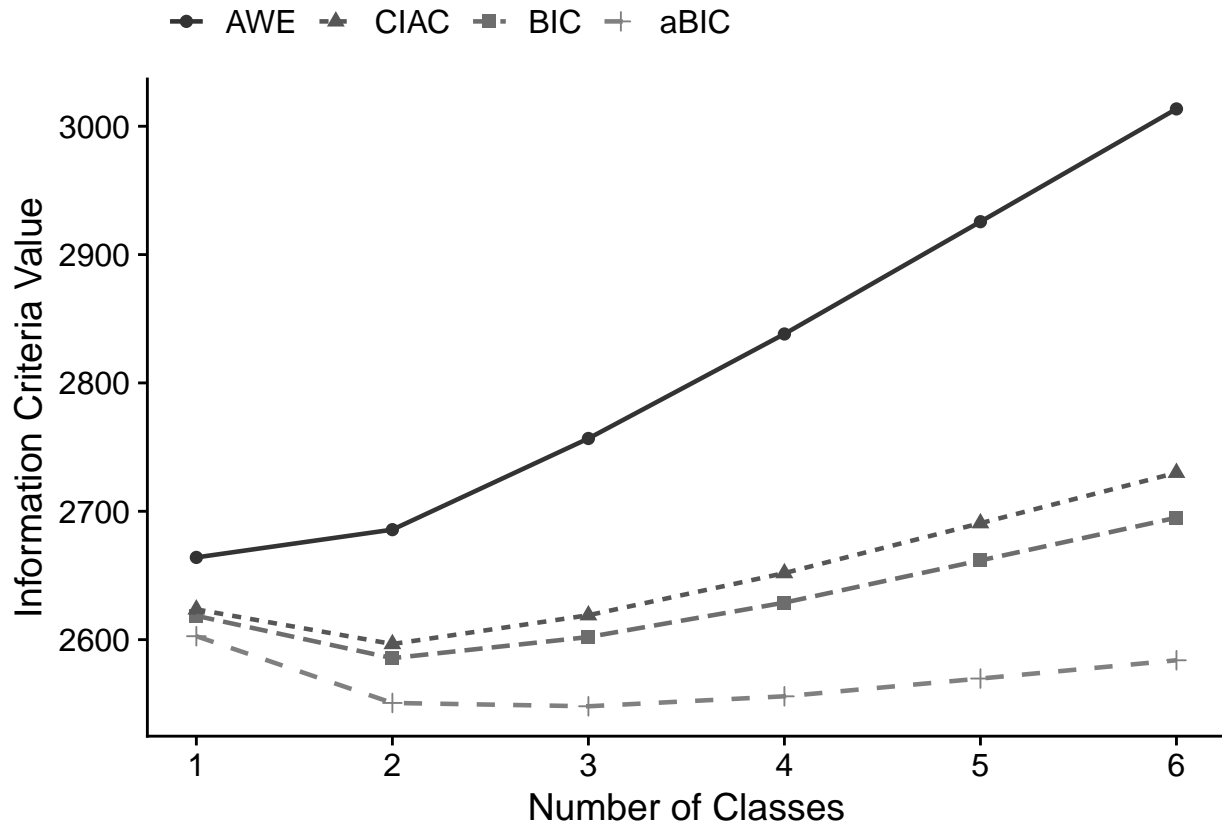
Save plot in the `figures` folder.

```
ggsave(here("figures", "Fig2_IC_plot.png"), dpi=300, height=5, width=7, units="in")
```

1.4 Compare Conditional Item Probability Plots

This syntax produces a plot containing a series of LCA probability plot facets. This can be used to compare solutions across K-class models.

```
model_results <- data.frame()

for (i in 1:length(output_enum)) {
  temp <- output_enum[[i]]$parameters$unstandardized
  temp <- data.frame(unclass(temp)) %>%
    mutate(model = paste0(i, "-Class Model"))
  model_results <- rbind(model_results, temp)
}

model_results <- model_results %>%
```
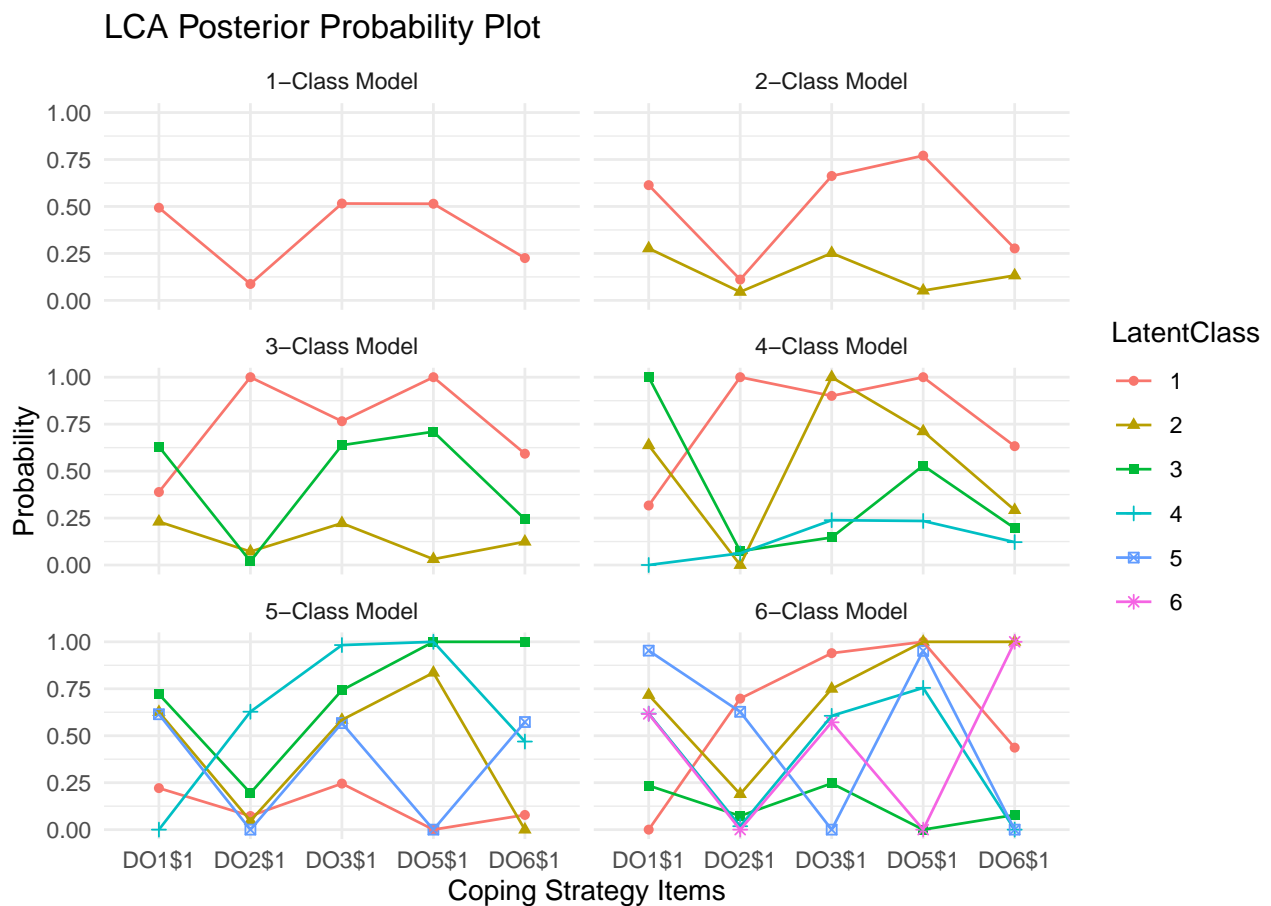
```
  filter(paramHeader == "Thresholds") %>%
  select(est, model, LatentClass, param) %>%
  mutate(prob = (1 / (1 + exp(est))))

ggplot(model_results,
       aes(x = param, y = prob,
           color = LatentClass,
           shape = LatentClass,
           group = LatentClass)) +
  geom_point() + geom_line() +
  facet_wrap(~ model, ncol = 2) +
  labs(title = "LCA Posterior Probability Plot",
       x= "Coping Strategy Items", y = "Probability") +
  theme_minimal()
```


LCA Posterior Probability Plot

```
ggsave(here("figures","FigA1_compare_kclass_LCAs.png"), dpi=300, height=4, width=6, units="in")
```

1.5 Plot Final Model - Conditional Item Probability Plot

This code generates the figure as presented in the manuscript for the final unconditional LCA model.

_____

This syntax creates a function called `plot_lca_function` that requires 7 arguments (inputs):

- `model_name`: name of Mplus model object (e.g., `model_step1`)
- `item_num`: the number of items in LCA measurement model (e.g., 5)
- `class_num`: the number of classes (k) in LCA model (e.g., 3)
- `item_labels`: the item labels for x-axis (e.g., `c("Enjoy","Useful","Logical","Job","Adult")`)
- `class_labels`: the class label names (e.g., `c("Adaptive Coping","Externalizing Behavior","No Coping")`)
- `class_legend_order` = change the order that class names are listed in the plot legend (e.g., `c(2,1,3)`)
- `plot_title`: include the title of the plot here (e.g., `"LCA Posterior Probability Plot"`)

Read in plot data from Mplus output file `Step1_3step_BD.out`

```r
model_step1 <- readModels(here("3step_mplus", "Step1_3step_BD.out"), quiet = TRUE)
```

```r
plot_lca_function <- function(model_name,item_num,class_num,item_labels,
                              class_labels,class_legend_order,plot_title){

mplus_model <- as.data.frame(model_name$gh5$means_and_variances_data$estimated_probs$values)
plot_data <- mplus_model[seq(2, 2*item_num, 2),]

c_size <- as.data.frame(model_name$class_counts$modelEstimated$proportion)
colnames(c_size) <- paste0("cs")
c_size <- c_size %>% mutate(cs = round(cs*100, 2))
colnames(plot_data) <- paste0(class_labels, glue(" ({c_size[1:class_num,]}%)"))
plot_data <- plot_data %>% relocate(class_legend_order)

plot_data <- cbind(Var = paste0("U", 1:item_num), plot_data)
plot_data$Var <- factor(plot_data$Var,
              labels = item_labels)
plot_data$Var <- fct_inorder(plot_data$Var)

pd_long_data <- melt(plot_data, id.vars = "Var")

# This syntax uses the date-frame created above to produce the plot with `ggplot()`

p <- pd_long_data %>%
  ggplot(aes(x = as.integer(Var), y = value,
  shape = variable, colour = variable, lty = variable)) +
  geom_point(size = 4) + geom_line() +
  scale_x_continuous("", breaks = 1:5, labels = plot_data$Var) +
  scale_colour_grey() +
  labs(title = plot_title, y = "Probability") +
  theme_cowplot() +
  theme(legend.title = element_blank(),
        legend.position = "top")


p
return(p)
}
```
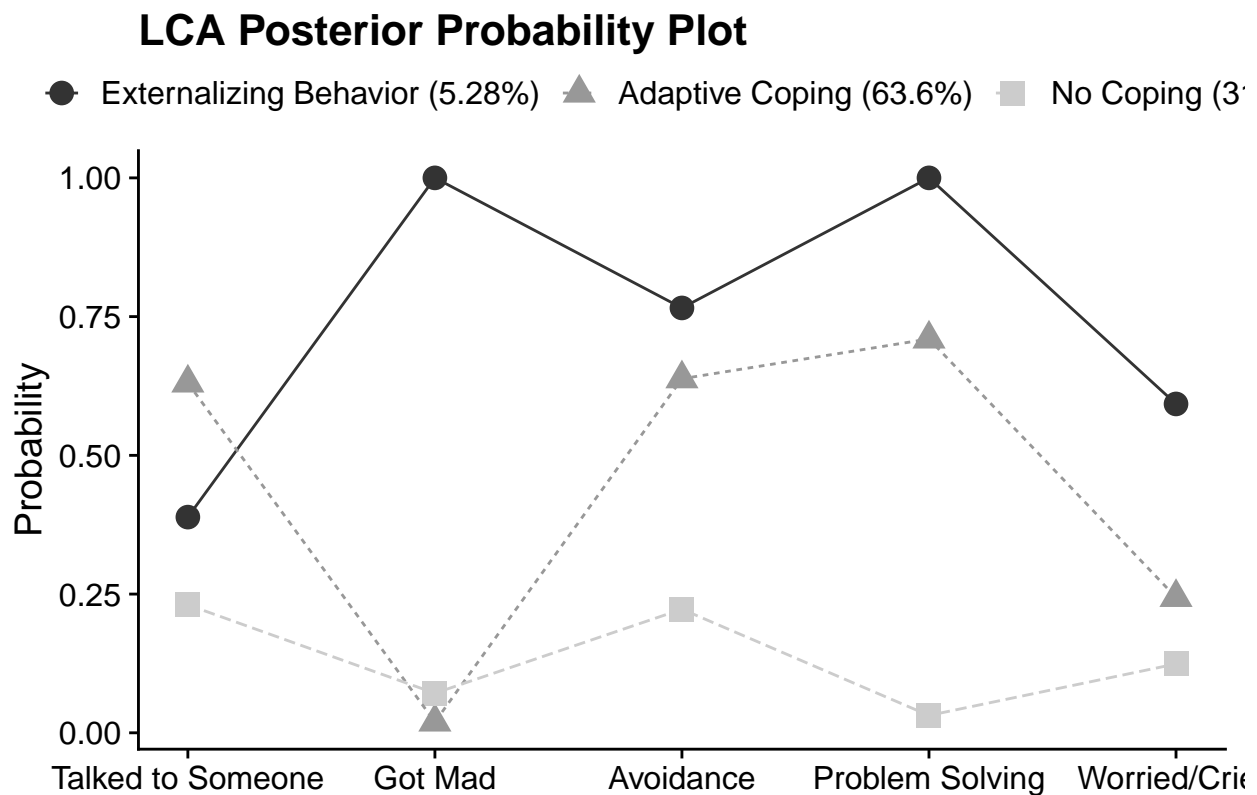
Run the `plot_lca_function` by specifying each input (*Figure 1*)

```
plot_lca_function(
  model_name = model_step1,
  item_num = 5,
  class_num = 3,
  item_labels = c("Talked to Someone","Got Mad", "Avoidance","Problem Solving", "Worried/Cried"),
  class_labels = c("Adaptive Coping","Externalizing Behavior","No Coping"),
  class_legend_order = c(2,1,3),
  plot_title = "LCA Posterior Probability Plot"
  )
```



```
ggsave(here("figures", "Fig1_LCA_3-Class.png"), dpi=300, height=5, width=7, units="in")
```

_____

1.6 Manual "3-Step" ML Auxiliary Variable Integration Method

_____

Step 1 - Estimate the unconditional model with all covariate & distal outcome variables mentioned in the `auxiliary` statement.

```
m_step1  <- mplusObject(
  TITLE = "Step1_3step_automation Behavioral Disorder",
  VARIABLE =
   "categorical = do1 do2 do3 do5 do6;

    usevar = do1 do2 do3 do5 do6;

    classes = c(3);

    !!! NOTE: All auxiliary variables to be considered in the final model should be listed here !!!
    auxiliary =
    FEMALE ETHN_CMP SOC_STRS
    BOTHR_U negmood1 posmood1;",

  ANALYSIS =
   "estimator = mlr;
    type = mixture;
    starts = 500 100;",

  SAVEDATA =
   "!!! NOTE: This saved dataset will contain class probabilities and modal assignment columns !!!
    File=3step_BD_savedata_012020.dat;
    Save=cprob;
    Missflag= 999;",

  MODEL = "",
  OUTPUT = "",

  PLOT =
    "type = plot3;
    series = do1 do2 do3 do5 do6(*);",

  usevariables = colnames(ies_data),
  rdata = ies_data)

m_step1_fit <- mplusModeler(m_step1,
                dataout=here("3step_mplus", "Step1_3step_BD.dat"),
                modelout=here("3step_mplus", "Step1_3step_BD.inp") ,
                check=TRUE, run = TRUE, hashfilename = FALSE)
```

_____

Step 2 - Extract logits & saved data from the step 1 unconditional model.

Extract logits for the classification probabilities for the most likely latent class

```
logit_cprobs <- as.data.frame(m_step1_fit[["results"]]
                                          [["class_counts"]]
                                          [["logitProbs.mostLikely"]])
```

Extract saved data from the step 1 model `mplusObject` named "m_step1_fit"

```
savedata <- as.data.frame(m_step1_fit[["results"]]
                                     [["savedata"]])
```

Rename the column in `savedata` for "C" and change to "N"

```
colnames(savedata)[colnames(savedata)=="C"] <- "N"
```

Step 3 (part 1) - Estimate the unconditional model with logits from step 2.

This model is estimated to check that the class proportions are approximately the same as in step 1.

```
m_step2  <- mplusObject(
  TITLE = "Step2_3step_automation Behavioral Disorder",

  VARIABLE =
 "nominal=N;
  USEVAR = n;
  missing are all (999);
  classes = c(3); ",

  ANALYSIS =
 "estimator = mlr;
  type = mixture;
  starts = 0;",

  MODEL =
    glue(
 "%C#1%
  [n#1@{logit_cprobs[1,1]}];
  [n#2@{logit_cprobs[1,2]}];

  %C#2%
  [n#1@{logit_cprobs[2,1]}];
  [n#2@{logit_cprobs[2,2]}];

  %C#3%
  [n#1@{logit_cprobs[3,1]}];
  [n#2@{logit_cprobs[3,2]}];"),

  OUTPUT = "!tech11  tech14 res;",
```

```
  PLOT =
 "!type = plot3;
  !series = do1 do2 do3 do5 do6(*);",

  usevariables = colnames(savedata),
  rdata = savedata)

m_step2_fit <- mplusModeler(m_step2,
                 dataout=here("3step_mplus", "Step2_3step_BD.dat"),
                 modelout=here("3step_mplus", "Step2_3step_BD.inp"),
                 check=TRUE, run = TRUE, hashfilename = FALSE)
```

_____

Step 3 (part 2) - Add covariates & distal outcomes to the model.

1.7 Moderation - Estimate the final SEM Model

_____

Specification details:

- This example contains two distal outcomes (`POSMOOD1` & `NEGMOOD1`) and one binary covariate (`SOC_STRS`).
- Under each class-specific statement (e.g., `%C#1%`) the distal outcomes are mentioned to estimate the intercept parameters.
- Moderation is specified by mentioning the `"outcome ON covariate;"` syntax under each of the class-specific statements.
- Note that the binary covariate is centered so that reported distal means (intercepts) are estimated at the weighted average of Social Stress.

```
m_step3  <- mplusObject(
  TITLE = "Step3_3step_automation Behavioral Disorder",

  VARIABLE =
 "nominal = N;
  usevar = n;
  missing are all (999);

  usevar = SOC_STRS POSMOOD1 NEGMOOD1;
  classes = c(3); ",

  DEFINE =
 "Center SOC_STRS (Grandmean);",

  ANALYSIS =
 "estimator = mlr;
  type = mixture;
  starts = 0;",

  MODEL =
```

```
  glue(
"!DISTAL = POSMOOD1 NEGMOOD1
 !MODERATOR = SOC_STRS

 %OVERALL%
 POSMOOD1 on SOC_STRS;
 POSMOOD1;

 NEGMOOD1 on SOC_STRS;
 NEGMOOD1;

 %C#1%
 [n#1@{logit_cprobs[1,1]}];
 [n#2@{logit_cprobs[1,2]}];

 [NEGMOOD1](m01);
 NEGMOOD1;                       !!! estimate conditional intercept !!!
 NEGMOOD1 on SOC_STRS (s01);    !!! estimate conditional regression !!!

 [POSMOOD1] (m1);
 POSMOOD1;
 POSMOOD1 on SOC_STRS (s1);

 %C#2%
 [n#1@{logit_cprobs[2,1]}];
 [n#2@{logit_cprobs[2,2]}];

 [NEGMOOD1](m02);
 NEGMOOD1;
 NEGMOOD1 on SOC_STRS (s02);

 [POSMOOD1] (m2);
 POSMOOD1;
 POSMOOD1 on SOC_STRS (s2);

 %C#3%
 [n#1@{logit_cprobs[3,1]}];
 [n#2@{logit_cprobs[3,2]}];

 [NEGMOOD1](m03);
 NEGMOOD1;
 NEGMOOD1 on SOC_STRS (s03);

 [POSMOOD1] (m3);
 POSMOOD1;
 POSMOOD1 on SOC_STRS (s3);"),

 MODELCONSTRAINT =
"New (diff12 diff13
 diff23 slope12 slope13
 slope23 ndiff12 ndiff13
 ndiff23 nslope12 nslope13
 nslope23);
```

```
   diff12 = m1-m2;    ndiff12 = m01-m02;
   diff13 = m1-m3;    ndiff13 = m01-m03;
   diff23 = m2-m3;    ndiff23 = m02-m03;
   slope12 = s1-s2;  nslope12 = s01-s02;
   slope13 = s1-s3;  nslope13 = s01-s03;
   slope23 = s2-s3;  nslope23 = s02-s03;",

   MODELTEST =
   ## NOTE: Only a single Wald test can be conducted per model run. Therefore,
   ## this example requires running separate models for each omnibus test (e.g.,
   ## 4 models; 2 outcomes and 2 slope coefficients). This can be done by
   ## commenting out all but one test and then making multiple input/output files.

  "m1=m2;        !!! Distal outcome omnibus Wald test for `POSMOOD1` !!!
   m2=m3;

   !s1=s2;       !!! Slope difference omnibus Wald test `POSMOOD1 on SOC_STRS` !!!
   !s2=s3;

   !m01=m02;     !!! Distal outcome omnibus Wald test for `NEGMOOD1` !!!
   !m02=m03;

   !s01=s02;   !!! Slope difference omnibus Wald test for `POSMOOD1 on SOC_STRS` !!!
   !s02=s03;",

   usevariables = colnames(savedata),
   rdata = savedata)

m_step3_fit <- mplusModeler(m_step3,
                 dataout=here("3step_mplus", "Step3_3step_BD.dat"),
                 modelout=here("3step_mplus", "Step3_3step_BD.inp"),
                 check=TRUE, run = TRUE, hashfilename = FALSE)
```

End of 3-step procedure

_____

Estimate step 3 model with covariate un-centered for simple-slopes plots.   This puts the intercepts fixed at the first level of the covariate (SOC_STRS). These values are needed to produce the simple slope plots.

Note: Here the update() function is used to take the previous model and remove the Mplus syntax within the DEFINE statement that was used to center the covariate Social Stress. Next, the updated model input syntax is used to estimate a new model. To learn more about the update function see the MplusAutomation tutorial article (https://www.tandfonline.com/doi/pdf/10.1080/10705511.2017.1402334).

```
m_uncen <- update(m_step3,
  DEFINE = ~" ") # This update removes the centering syntax from the model object `m_step3`

m_uncen_fit <- mplusModeler(m_uncen,
    dataout=here("3step_mplus", "Step3_3step_BD.dat"),
    modelout=here("3step_mplus", "Step3_uncentered_BD.inp"),
    check=TRUE, run = TRUE, hashfilename = FALSE)UCSB_Navy_mark.png
```

1.8 Distal Outcome Plot

Note: The distal outcome means are estimated with the binary covariate (`SOC_STRS`) at the weighted average. This is specified by centering social stress as shown in the `Step-3` model syntax.

This syntax reads in the `Step3` model & extract model parameter estimates.

```
model_step3 <- readModels(here("3step_mplus", "Step3_3step_BD.out"), quiet = TRUE)

model_step3 <- data.frame(model_step3$parameters$unstandardized)
```

This syntax is used to create the `data-frame` that produces the distal outcome bar plot.

```
distal_data <- model_step3 %>%
  filter(paramHeader == "Intercepts") %>%
  mutate(param = case_when(
      param == "POSMOOD1" ~ "Positive Mood",
      param == "NEGMOOD1" ~ "Negative Mood")) %>%
  mutate(LatentClass = factor(LatentClass,
      labels = c("Adaptive Coping", "Externalizing Behavior","No Coping"))) %>%
  mutate(value_labels = c("3.16 a", "1.82 d", "2.67 b", "2.31 c", "2.72 b", "1.67 d"))
```

Plot distal outcomes grouped by class (*Figure 3*)

```
ggplot(distal_data,
       aes(fill=LatentClass, y=est, x=fct_rev(param))) +
  geom_bar(position="dodge", stat="identity", color="black", size=.3) +
  geom_errorbar( aes(ymin=est-se, ymax=est+se),
                 width=.2, position=position_dodge(.9)) +
  geom_text(aes(y = est -.5, label = value_labels),
            family="Times New Roman", size=4,
            position=position_dodge(.9)) +
  scale_fill_grey(start = 0.6, end = 1.0) +
  theme_cowplot() +
  xlab("") + ylab("Mean Value") +
  theme(text=element_text(family="Times New Roman", size=12),
        axis.text.x=element_text(size=10)) +
        coord_cartesian(expand = FALSE)
```
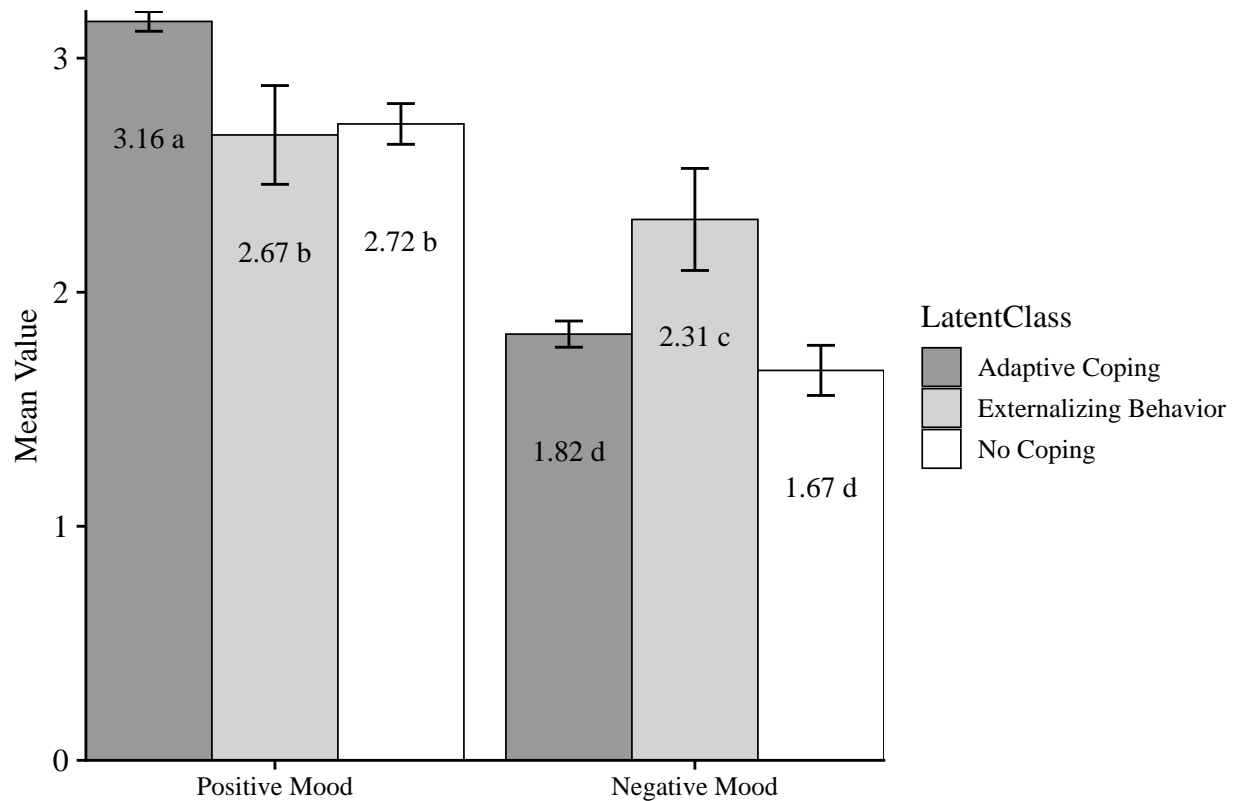
```r
ggsave(here("figures","Fig3_distal_barplot.png"), dpi=300, height=4, width=6, units="in")
```

1.9 Simple Slope Plots

Note: The un-centered distal intercepts represent the conditional means when the binary covariate is at its first level `SOC_STRS = 0` (i.e., no social stress). Therefore, the conditional mean for `SOC_STRS = 1` (i.e., social stress) can be calculated by adding the associated slope coefficient to the intercept.

Read in the un-centered model & extract relevant parameters

```r
model_uncen <- readModels(here("3step_mplus", "Step3_uncentered_BD.out"), quiet = TRUE)

model_uncen <- data.frame(model_uncen$parameters$unstandardized)

slope_data <- model_uncen %>%
  filter(str_detect(paramHeader, 'ON|Inter')) %>%
  unite("param", paramHeader:param, remove = TRUE) %>%
  mutate(param = str_replace(param, "MOOD1.ON_SOC_STRS", "_COEF")) %>%
  mutate(param = str_remove_all(param, "Intercepts_|MOOD1")) %>%
  mutate(LatentClass = factor(LatentClass,
    labels = c("Adaptive Coping (63.6%)", "Externalizing Behavior (5.3%)","No Coping (31.1%)")))
```

16

Positive mood simple slope graph

Prepare `data-frame` for plotting

```
pos_data <- slope_data %>%
  filter(str_detect(param, 'POS'))

pos_wide <- pos_data %>%
  select(param,est, LatentClass) %>%
  pivot_wider(names_from = param, values_from = est) %>%
  rename("No.Social.Stress" = 'POS') %>%
  mutate(Social.Stress = No.Social.Stress + POS_COEF) %>% # calc. condit. means `SOC_STRS = 1`
  select(-POS_COEF)

plot_pos <- melt(pos_wide, id.vars = "LatentClass") %>%
  mutate(variable = factor(variable,
                    levels = c("No.Social.Stress","Social.Stress"),
                    labels = c("No Social Stress","Social Stress")))
```

Plot positive mood simple slope graph

```
p_plot <- ggplot(plot_pos,
          aes(y=value, x=variable,
              color=LatentClass,
              group=LatentClass,
              shape=LatentClass,
              lty=LatentClass)) +
  geom_point(size = 4) + geom_line() +
  xlab("") + ylab("Positive Mood")  + ylim(2.5,3.5)+
  scale_colour_grey() +
  theme_classic() +
  theme(text=element_text(family="Times New Roman", size=12),
      axis.text.x=element_text(size=12),
      legend.text = element_text(family="Times New Roman", size=10),
      legend.position = "top", legend.title = element_blank()) +
    annotate(geom = "text",
        x = 1.8, y = 2.77,
        label = "N.S.", color = "black") +
    annotate(geom = "text",
        x = 1.8, y = 2.60,
        label = "N.S.", color = "black")
```

Negative mood simple slope graph

Prepare `data-frame` for plotting

```r
neg_data <- slope_data %>%
  filter(str_detect(param, 'NEG'))

neg_wide <- neg_data %>%
  select(param,est, LatentClass) %>%
  pivot_wider(names_from = param, values_from = est) %>%
  rename("No.Social.Stress" = 'NEG') %>%
  mutate(Social.Stress = No.Social.Stress + NEG_COEF) %>%   # calculate means for `SOC_STRS = 1`
  select(-NEG_COEF)

plot_neg <- melt(neg_wide, id.vars = "LatentClass") %>%
  mutate(variable = factor(variable,
                    levels = c("No.Social.Stress","Social.Stress"),
                    labels = c("No Social Stress","Social Stress")))
```

Plot negative mood simple slope graph

```r
n_plot <- ggplot(plot_neg,
           aes(y=value, x=variable,
           color=LatentClass,
           group=LatentClass,
           shape=LatentClass,
           lty=LatentClass)) +
  geom_point(size=4) + geom_line() +
  xlab("") + ylab("Negative Mood") + ylim(1.5,3)+
  scale_colour_grey() + theme_classic() +
  theme(text=element_text(family="Times New Roman", color = "black", size=12),
        axis.text.x=element_text(size=12),
        legend.text = element_text(family="Times New Roman", size=10),
        legend.position = "top", legend.title = element_blank()) +
    annotate(geom = "text",
       x = 1.8, y = 1.6,
       label = "N.S.", color = "black")
```

Combine the two simple slopes graphs for distal outcomes positive & negative mood (*Figure 5*)

Note: This code combines 2 ggplot objects using the {patchwork} package.

```
p_plot / n_plot
```
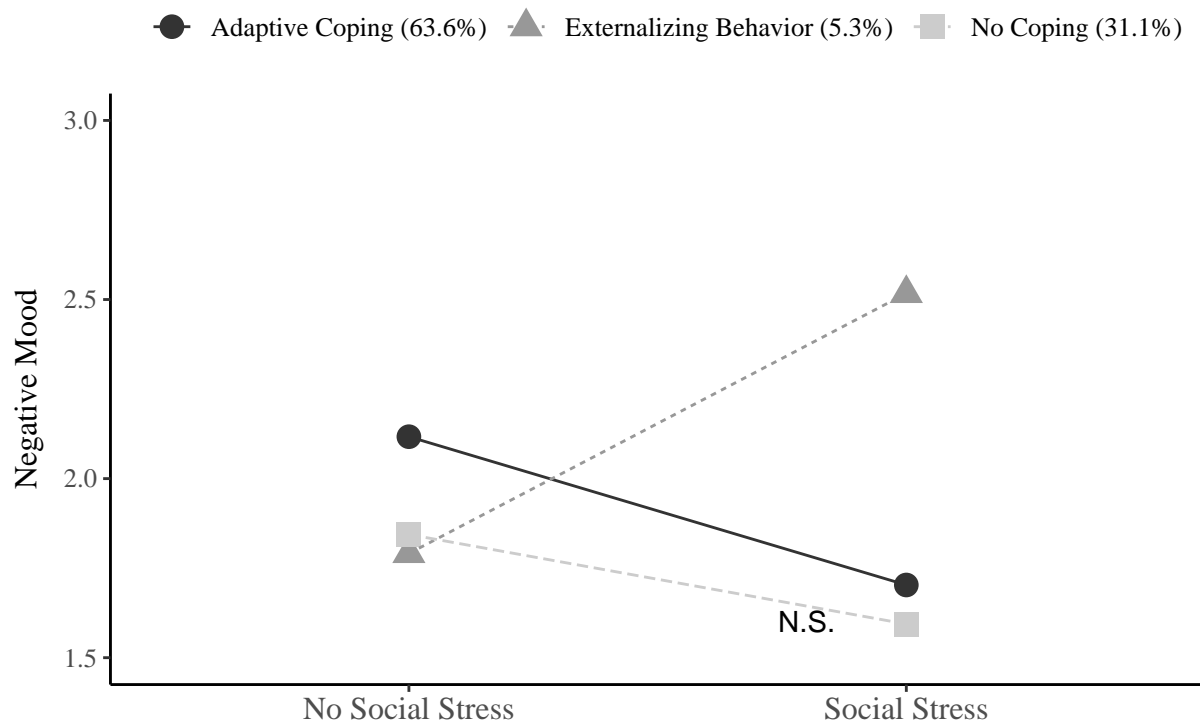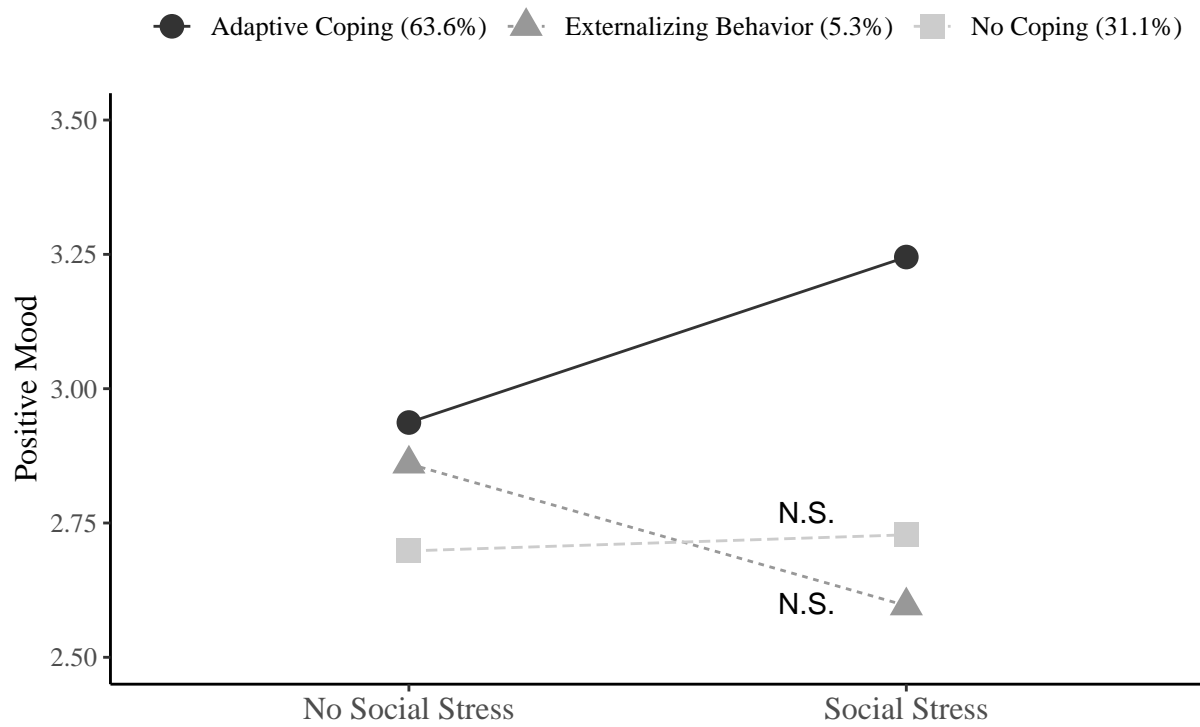


```
ggsave(here("figures", "Fig5_simple_slopes.png"), dpi=300, height=8.5, width=6.5, units="in")
```

2.0 Response Pattern Table

To replicate the response pattern table produced in this study (*Table 3*) see the associated `R/MplusAutomation` code example here:

https://garberadamc.github.io/project-site/Lab10.1-response-patterns

Additional examples using `R/MplusAutomation` for a range of analyses can be found here:

https://garberadamc.github.io/project-site

References:

Hallquist, Michael N., and Joshua F. Wiley. 2018. "MplusAutomation: An R Package for Facilitating Large-Scale Latent Variable Analyses in Mplus." Structural Equation Modeling, 1–18. https://doi.org/10.1080/10705511.2017.1402334.

Miller, Jon D. Longitudinal Study of American Youth (LSAY), Seventh Grade Data, 1987-1988; 2015-2016. Ann Arbor, MI: Inter-university Consortium for Political and Social Research [distributor], 2019-04-23. https://doi.org/10.3886/ICPSR37287.v1

Müller, Kirill. 2017.Here: A Simpler Way to Find Your Files. https://CRAN.R-project.org/package=here.

Muthen L.K., & Muthen B.O. (1998-2017) Mplus User's Guide. Eight Edition. Los Angelos, CA: Muthen & Muthen.

R Core Team. 2019.R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Wickham H, et al. (2019). "Welcome to the tidyverse." Journal of Open Source Software, 4(43), 1686. doi: 10.21105/joss.01686.