Package ‘MicroMacroMultilevel’

July 21, 2025

Type Package

Description Most multilevel methodologies can only model macro-micro multilevel situations in an unbiased way, wherein group-level predictors (e.g., city temperature) are used to predict an individual-level outcome variable (e.g., citizen personality). In contrast, this R package enables researchers to model micro-macro situations, wherein individual-level (micro) predictors (and other group-level predictors) are used to predict a group-level (macro) outcome variable in an unbiased way.

Title Micro-Macro Multilevel Modeling

Version 0.4.0

License GPL (>= 2)

Depends R (>= 3.1.0)

RoxygenNote 6.0.1

LazyData TRUE

NeedsCompilation no

Author Jackson G Lu [aut],
Elizabeth Page-Gould [aut],
Nancy R Xu [aut, cre]

Maintainer Nancy R Xu <nancyranxu@gmail.com>

Repository CRAN

Date/Publication 2017-07-01 14:47:20 UTC

Contents

`adjusted.predictors` *Calculating the Adjusted Group Means of Individual-Level Variables in a Micro-Macro Multilevel Situation*

Description

As the prerequisite step of fitting a micro-macro multilevel model, this function calculates the adjusted group means of individual-level predictors in an unbiased way.

Usage

```
adjusted.predictors(x.data, z.data, x.gid, z.gid)
```

Arguments

`x.data`	an N-by-p data frame of individual-level predictors, where N denotes the total number of individuals and p denotes the number of individual-level predictors. Must contain no NAs.
`z.data`	a G-by-q data frame of group-level predictors, where G denotes the total number of groups and q denotes the number of group-level predictors. Must contain no NAs.
`x.gid`	an array or an N-by-1 numeric matrix of each individual's group ID. The order corresponds to the individuals in x.data. Duplicates expected.
`z.gid`	an array or a G-by-1 numeric matrix of Group ID. The order corresponds to the groups in z.data. All group IDs should be unique (i.e., no duplicates allowed).

Details

To date, most multilevel methodologies can only unbiasedly model macro-micro multilevel situations, wherein group-level predictors (e.g., city temperature) are used to predict an individual-level outcome variable (e.g., citizen personality). In contrast, this R package enables researchers to model micro-macro situations, wherein individual-level (micro) predictors (and other group-level predictors) are used to predict a group-level (macro) outcome variable in an unbiased way.

To conduct micro-macro multilevel modeling with the current package, one must first compute the adjusted group means with the function `adjusted.predictors`. This is because in micro-macro multilevel modeling, it is statistically biased to directly regress the group-level outcome variable on the unadjusted group means of individual-level predictors (Croon & van Veldhoven, 2007). Instead, one should use the best linear unbiased predictors (BLUP) of the group means (i.e., the adjusted group means), which is conveniently computed by `adjusted.predictors`.

Once produced by `adjusted.predictors`, the adjusted group means can be used as one of the inputs of the `micromacro.lm` function, which reports estimation results and inferential statistics of the micro-macro multilevel model of interest. Importantly, `adjusted.predictors` also reports whether group size is the same across all groups, which is a critical dummy input of the `micromacro.lm` function.

Value

adjusted.group.means a G-by-(p+q+1) numeric matrix that contains p adjusted group means of the individual-level variables from x.data, q group-level predictors from z.data, and unique group IDs.

unequal.groups a boolean variable. TRUE = group size is different across groups; FALSE = group size is the same across groups.

group.size a G-by-2 data frame that displays unique group IDs and the corresponding group sizes.

Author(s)

Jackson G. Lu, Elizabeth Page-Gould, Nancy R. Xu (maintainer, nancyranxu@gmail.com).

References

Akinola, M., Page-Gould, E., Mehta, P. H., & Lu, J. G. (2016). Collective hormonal profiles predict group performance. Proceedings of the National Academy of Sciences, 113 (35), 9774-9779.

Croon, M. A., & van Veldhoven, M. J. (2007). Predicting group-level outcome variables from variables measured at the individual level: A latent variable multilevel model. Psychological Methods, 12(1), 45-57.

See Also

micromacro.lm for fitting the micro-macro multilevel linear model of interest.

Examples

```
######## SETUP: DATA GENERATING PROCESSES ########
set.seed(123)
# Step 1. Generate a G-by-q data frame of group-level predictors (e.g., control variables), z.data
# In this example, G = 40, q = 2
group.id = seq(1, 40)
z.var1 = rnorm(40, mean=0, sd=1)
z.var2 = rnorm(40, mean=100, sd=2)
z.data = data.frame(group.id, z.var1, z.var2)
# Step 2. Generate a G-by-p data frame of group-level means for the predictors that will be used to
# generate x.data
# In this example, there are 3 individual-level predictors, thus p = 3
x.var1.means = rnorm(40, mean=50, sd = .05)
x.var2.means = rnorm(40, mean=20, sd = .05)
x.var3.means = rnorm(40, mean=-10, sd = .05)
x.data.means = data.frame(group.id, x.var1.means, x.var2.means, x.var3.means)
# Step 3. Generate two N-by-p data frames of individual-level predictors, x.data
# One of these two data frames assumes unequal-sized groups (Step 3a),
# whereas the other assumes equal-sized groups (Step 3b):
# Step 3a. Generate the individual-level predictors
# In this example, N = 200 and group size is unequal
x.data.unequal = data.frame( group.id=rep(1:40, times=sample( c(4,5,6), 40, replace=TRUE) )[1:200] )
x.data.unequal = merge( x.data.unequal,
         data.frame( group.id, x.var1.means, x.var2.means, x.var3.means ), by="group.id" )
x.data.unequal = within( x.data.unequal, {
  x.var1 = x.var1.means + rnorm(200, mean=0, sd = 2)
```

```
  x.var2 = x.var2.means + rnorm(200, mean=0, sd = 6)
  x.var3 = x.var3.means + rnorm(200, mean=0, sd = 1.5)
})
# Step 3b. Generate the individual-level predictors
# In this example, N = 200 and group size is equal
x.data.equal = data.frame( group.id=rep(1:40, each=5) )
x.data.equal = merge( x.data.equal, x.data.means, by="group.id" )
x.data.equal = within( x.data.equal, {
  x.var1 = x.var1.means + rnorm(200, mean=0, sd = 2)
  x.var2 = x.var2.means + rnorm(200, mean=0, sd = 6)
  x.var3 = x.var3.means + rnorm(200, mean=0, sd = 1.5)
})
# Step 3. Generate a G-by-1 data frame of group-level outcome variable, y
# In this example, G = 40
y = rnorm(40, mean=6, sd=5)

apply(x.data.equal,2,mean)
#    group.id x.var1.means x.var2.means x.var3.means      x.var3      x.var2      x.var1
# 20.500000    50.000393    19.994708    -9.999167   -10.031995    20.185361    50.084635
apply(x.data.unequal,2,mean)
#    group.id x.var1.means x.var2.means x.var3.means      x.var3      x.var2      x.var1
# 20.460000    50.002286    19.994605    -9.997034    -9.983146    19.986111    50.123591
apply(z.data,2,mean)
# z.var1      z.var2
# 0.04518332 99.98656817
mean(y)
# 6.457797

######## EXAMPLE 1. GROUP SIZE IS DIFFERENT ACROSS GROUPS ########
######## Need to use adjusted.predictors() in the same package ###

# Step 4a. Generate a G-by-1 matrix of group ID, z.gid. Then generate an N-by-1 matrix of
# each individual's group ID, x.gid, where the group sizes are different
z.gid = seq(1:40)
x.gid = x.data.unequal$group.id
# Step 5a. Generate the best linear unbiased predictors that are calcualted from
# individual-level data
x.data = x.data.unequal[,c("x.var1","x.var2","x.var3")]
results = adjusted.predictors(x.data, z.data, x.gid, z.gid)
# Note: Given the fixed random seed, the output should be as below
results$unequal.groups
# TRUE
names(results$adjusted.group.means)
# "BLUP.x.var1" "BLUP.x.var2" "BLUP.x.var3" "z.var1"      "z.var2"      "gid"
head(results$adjusted.group.means)
#   BLUP.x.var1 BLUP.x.var2 BLUP.x.var3 group.id      z.var1    z.var2 gid
# 1    50.05308    20.83911  -10.700361        1 -0.56047565  98.61059   1
# 2    48.85559    22.97411   -9.957270        2 -0.23017749  99.58417   2
# 3    50.16357    19.50001   -9.645735        3  1.55870831  97.46921   3
# 4    49.61853    21.25962  -10.459398        4  0.07050839 104.33791   4
# 5    50.49673    21.38353   -9.789924        5  0.12928774 102.41592   5
# 6    50.86154    19.15901   -9.245675        6  1.71506499  97.75378   6
```

```
######## EXAMPLE 2. GROUP SIZE IS THE SAME ACROSS ALL GROUPS ########
######## Need to use adjusted.predictors() in the same package ###

# Step 4b. Generate a G-by-1 matrix of group ID, z.gid. Then generate an N-by-1 matrix of
# each individual's group ID, x.gid, where group size is the same across all groups
z.gid = seq(1:40)
x.gid = x.data.equal$group.id
# Step 5b. Generate the best linear unbiased predictors that are calcualted from
# individual-level data
x.data = x.data.equal[,c("x.var1","x.var2","x.var3")]
results = adjusted.predictors(x.data, z.data, x.gid, z.gid)
results$unequal.groups
# FALSE
names(results$adjusted.group.means)
# "BLUP.x.var1" "BLUP.x.var2" "BLUP.x.var3" "z.var1"      "z.var2"      "gid"
results$adjusted.group.means[1:5, ]
#   BLUP.x.var1 BLUP.x.var2 BLUP.x.var3 group.id      z.var1    z.var2 gid
# 1    50.91373    19.12994  -10.051647        1 -0.56047565  98.61059   1
# 2    50.19068    19.17978  -10.814382        2 -0.23017749  99.58417   2
# 3    50.13390    20.98893   -9.952348        3  1.55870831  97.46921   3
# 4    49.68169    19.60632  -10.612717        4  0.07050839 104.33791   4
# 5    50.28579    22.07469  -10.245505        5  0.12928774 102.41592   5
```

---

micromacro.lm *Fitting Micro-Macro Multilevel Linear Models*

Description

After computing the adjusted group means of individual-level predictors by `adjusted.predictors`, use `micromacro.lm` for estimation results and inferential statistics.

Usage

```
micromacro.lm(model, adjusted.predictors, y, unequal.groups = NULL)
```

Arguments

- `model` a linear regression model formula, e.g., as.formula(y ~ x1 + x2 ... + xm).
- `adjusted.predictors` a G-by-m data frame, where column variables are group-level predictors and the adjusted group means of individual-level predictors were computed by the `adjusted.predictors` function. G denotes the number of groups and m denotes the number of predictors in the model.
- `y` an array or a G-by-1 numeric matrix that corresponds to the group-level outcome variable in the model.
- `unequal.groups` an optional boolean variable automatically reported by the `adjusted.predictors` function. TRUE = group size is different across groups; FALSE = group size is the same across groups. Default is FALSE (same group size).

Details

To date, most multilevel methodologies can only unbiasedly model macro-micro multilevel situations, wherein group-level predictors (e.g., city temperature) are used to predict an individual-level outcome variable (e.g., citizen personality). In contrast, this R package enables researchers to model micro-macro situations, wherein individual-level (micro) predictors (and other group-level predictors) are used to predict a group-level (macro) outcome variable in an unbiased way.

To conduct micro-macro multilevel modeling with the current package, one must first compute the adjusted group means with the function `adjusted.predictors`. This is because in micro-macro multilevel modeling, it is statistically biased to directly regress the group-level outcome variable on the unadjusted group means of individual-level predictors (Croon & van Veldhoven, 2007). Instead, one should use the best linear unbiased predictors (BLUP) of the group means (i.e., the adjusted group means), which is conveniently computed by `adjusted.predictors`.

Once produced by `adjusted.predictors`, the adjusted group means can be used as one of the inputs of the `micromacro.lm` function, which reports estimation results and inferential statistics of the micro-macro multilevel model of interest.

If group size is the same across all groups (i.e., unequal.groups = FALSE), then OLS standard errors are reported and used to determine the inferential statistics in this micro-macro model. If group size is different across groups (i.e., unequal.groups = TRUE), however, then the heteroscedasticity-consistent standard errors are reported and used determine the inferential statistics in this micro-macro model (White, 1980).

Value

statistics a summary reports standard inferential statistics on linear regression, e.g., "Estimate", coefficient estimates; "Uncorrected S.E."/"S.E.", OLS standard errors; "Corrected S.E.", heteroskedasticity-consistent standard errors; "df", degree of freedom; "t", Student t statistics; "Pr(>|t|)", two-sided p-value; "r", effect size.

rsquared r squared.

rsquared.adjusted adjusted r squared.

residuals residuals from the model.

fitted.values fitted values from the model.

fstatistic F statistics of the model.

model.formula model formula.

Author(s)

Jackson G. Lu, Elizabeth Page-Gould, & Nancy R. Xu (maintainer, nancyranxu@gmail.com).

References

Akinola, M., Page-Gould, E., Mehta, P. H., & Lu, J. G. (2016). Collective hormonal profiles predict group performance. Proceedings of the National Academy of Sciences, 113 (35), 9774-9779.

Croon, M. A., & van Veldhoven, M. J. (2007). Predicting group-level outcome variables from variables measured at the individual level: A latent variable multilevel model. Psychological Methods, 12(1), 45-57.

White, H. (1980). A heteroskedasticity-consistent covariance estimator and a direct test of heteroskedasticity. Econometrica, 48, 817-838.

See Also

adjusted.predictors for calculating the adjusted group means of the individual-level predictors, and micromacro.summary for a friendly output summary table.

Examples

```
######## SETUP: DATA GENERATING PROCESSES ########
set.seed(123)
# Step 1. Generate a G-by-q data frame of group-level predictors (e.g., control variables), z.data
# In this example, G = 40, q = 2
group.id = seq(1, 40)
z.var1 = rnorm(40, mean=0, sd=1)
z.var2 = rnorm(40, mean=100, sd=2)
z.data = data.frame(group.id, z.var1, z.var2)
# Step 2. Generate a G-by-p data frame of group-level means for the predictors that will be used to
# generate x.data
# In this example, there are 3 individual-level predictors, thus p = 3
x.var1.means = rnorm(40, mean=50, sd = .05)
x.var2.means = rnorm(40, mean=20, sd = .05)
x.var3.means = rnorm(40, mean=-10, sd = .05)
x.data.means = data.frame(group.id, x.var1.means, x.var2.means, x.var3.means)
# Step 3. Generate two N-by-p data frames of individual-level predictors, x.data
# One of these two data frames assumes unequal-sized groups (Step 3a), whereas the other assumes
# equal-sized groups (Step 3b):
# Step 3a. Generate the individual-level predictors
# In this example, N = 200 and group size is unequal
x.data.unequal = data.frame( group.id=rep(1:40, times=sample( c(4,5,6), 40, replace=TRUE) )[1:200] )
x.data.unequal = merge( x.data.unequal,
           data.frame( group.id, x.var1.means, x.var2.means, x.var3.means ), by="group.id" )
x.data.unequal = within( x.data.unequal, {
  x.var1 = x.var1.means + rnorm(200, mean=0, sd = 2)
  x.var2 = x.var2.means + rnorm(200, mean=0, sd = 6)
  x.var3 = x.var3.means + rnorm(200, mean=0, sd = 1.5)
})
# Step 3b. Generate the individual-level predictors
# In this example, N = 200 and group size is equal
x.data.equal = data.frame( group.id=rep(1:40, each=5) )
x.data.equal = merge( x.data.equal, x.data.means, by="group.id" )
x.data.equal = within( x.data.equal, {
  x.var1 = x.var1.means + rnorm(200, mean=0, sd = 2)
  x.var2 = x.var2.means + rnorm(200, mean=0, sd = 6)
  x.var3 = x.var3.means + rnorm(200, mean=0, sd = 1.5)
})
# Step 3. Generate a G-by-1 data frame of group-level outcome variable, y
# In this example, G = 40
y = rnorm(40, mean=6, sd=5)

apply(x.data.equal,2,mean)
```

```
#    group.id x.var1.means x.var2.means x.var3.means      x.var3      x.var2      x.var1
# 20.500000    50.000393    19.994708    -9.999167   -10.031995    20.185361    50.084635
apply(x.data.unequal,2,mean)
#    group.id x.var1.means x.var2.means x.var3.means      x.var3      x.var2      x.var1
# 20.460000    50.002286    19.994605    -9.997034    -9.983146    19.986111    50.123591
apply(z.data,2,mean)
# z.var1      z.var2
# 0.04518332 99.98656817
mean(y)
# 6.457797

######## EXAMPLE 1. GROUP SIZE IS DIFFERENT ACROSS GROUPS ########
######## Need to use adjusted.predictors() in the same package ###

# Step 4a. Generate a G-by-1 matrix of group ID, z.gid. Then generate an N-by-1 matrix of
# each individual's group ID, x.gid, where the group sizes are different
z.gid = seq(1:40)
x.gid = x.data.unequal$group.id
# Step 5a. Generate the best linear unbiased predictors that are calcualted from
# individual-level data
x.data = x.data.unequal[,c("x.var1","x.var2","x.var3")]
results = adjusted.predictors(x.data, z.data, x.gid, z.gid)
# Note: Given the fixed random seed, the output should be as below
results$unequal.groups
# TRUE
names(results$adjusted.group.means)
# "BLUP.x.var1" "BLUP.x.var2" "BLUP.x.var3" "z.var1"      "z.var2"      "gid"
head(results$adjusted.group.means)
#   BLUP.x.var1 BLUP.x.var2 BLUP.x.var3 group.id      z.var1    z.var2 gid
# 1    50.05308    20.83911  -10.700361        1 -0.56047565  98.61059   1
# 2    48.85559    22.97411   -9.957270        2 -0.23017749  99.58417   2
# 3    50.16357    19.50001   -9.645735        3  1.55870831  97.46921   3
# 4    49.61853    21.25962  -10.459398        4  0.07050839 104.33791   4
# 5    50.49673    21.38353   -9.789924        5  0.12928774 102.41592   5
# 6    50.86154    19.15901   -9.245675        6  1.71506499  97.75378   6
# Step 6a. Fit a micro-macro multilevel model when group sizes are different
model.formula = as.formula(y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2)
model.output = micromacro.lm(model.formula, results$adjusted.group.means, y, results$unequal.groups)
micromacro.summary(model.output)
# Call:
#   micromacro.lm( y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2, ...)
#
# Residuals:
#        Min        1Q  Median       3Q      Max
# -13.41505 -2.974074 1.13077 3.566021 6.975819
#
#
# Coefficients:
#           Estimate  Uncorrected S.E. Corrected S.E. df          t  Pr(>|t|)          r
# (Intercept) 78.1232185    121.5103390  122.1367432 34  0.6396373 0.5266952 0.10904278
# BLUP.x.var1 -0.7589602      1.4954434    1.7177575 34 -0.4418320 0.6614084 0.07555696
# BLUP.x.var2  0.4263309      0.7070773    0.6299759 34  0.6767416 0.5031484 0.11528637
# BLUP.x.var3  0.2658078      2.4662049    2.4051691 34  0.1105152 0.9126506 0.01894980
```

```
# z.var1       0.4315941      1.0855707    1.0614535 34  0.4066068 0.6868451 0.06956356
# z.var2      -0.3949955      0.5573789    0.4230256 34 -0.9337390 0.3570228 0.15812040
#
# ---
#   Residual standard error: 5.1599 on 34 degrees of freedom
# Multiple R-squared: 0.0400727607, Adjusted R-squared: -0.1010930098
# F-statistic: 0.28387 on 5 and 34 DF, p-value: 0.91869

model.output$statistics
#            Estimate  Uncorrected S.E. Corrected S.E. df          t  Pr(>|t|)          r
# (Intercept) 78.1232185    121.5103390  122.1367432 34  0.6396373 0.5266952 0.10904278
# BLUP.x.var1 -0.7589602      1.4954434    1.7177575 34 -0.4418320 0.6614084 0.07555696
# BLUP.x.var2  0.4263309      0.7070773    0.6299759 34  0.6767416 0.5031484 0.11528637
# BLUP.x.var3  0.2658078      2.4662049    2.4051691 34  0.1105152 0.9126506 0.01894980
# z.var1       0.4315941      1.0855707    1.0614535 34  0.4066068 0.6868451 0.06956356
# z.var2      -0.3949955      0.5573789    0.4230256 34 -0.9337390 0.3570228 0.15812040
model.output$rsquared
# 0.0400727607
model.output$rsquared.adjusted
# -0.1010930098

######## EXAMPLE 2. GROUP SIZE IS THE SAME ACROSS ALL GROUPS ########
######## Need to use adjusted.predictors() in the same package ######

# Step 4b. Generate a G-by-1 matrix of group ID, z.gid. Then generate an N-by-1 matrix of
# each individual's group ID, x.gid, where group size is the same across all groups
z.gid = seq(1:40)
x.gid = x.data.equal$group.id
# Step 5b. Generate the best linear unbiased predictors that are calcualted from
# individual-level data
x.data = x.data.equal[,c("x.var1","x.var2","x.var3")]
results = adjusted.predictors(x.data, z.data, x.gid, z.gid)
results$unequal.groups
# FALSE
names(results$adjusted.group.means)
# "BLUP.x.var1" "BLUP.x.var2" "BLUP.x.var3" "z.var1"      "z.var2"      "gid"
results$adjusted.group.means[1:5, ]
#   BLUP.x.var1 BLUP.x.var2 BLUP.x.var3 group.id      z.var1    z.var2 gid
# 1    50.91373    19.12994  -10.051647        1 -0.56047565  98.61059   1
# 2    50.19068    19.17978  -10.814382        2 -0.23017749  99.58417   2
# 3    50.13390    20.98893   -9.952348        3  1.55870831  97.46921   3
# 4    49.68169    19.60632  -10.612717        4  0.07050839 104.33791   4
# 5    50.28579    22.07469  -10.245505        5  0.12928774 102.41592   5
# Step 6b. Fit a micro-macro multilevel model when group size is the same across groups
model.output2 = micromacro.lm(model.formula, results$adjusted.group.means, y,
                              results$unequal.groups)
micromacro.summary(model.output2)
# Call:
#   micromacro.lm( y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2, ...)
#
# Residuals:
#        Min        1Q    Median      3Q      Max
# -12.94409 -1.898937 0.8615494 3.78739 8.444582
```

```
#
#
# Coefficients:
#                Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept) 135.4109966 134.1478457 34  1.0094161 0.3199052 0.17057636
# BLUP.x.var1  -2.1984308   2.2203278 34 -0.9901379 0.3291012 0.16741080
# BLUP.x.var2  -0.6369600   0.8619558 34 -0.7389706 0.4649961 0.12572678
# BLUP.x.var3  -0.5121002   1.7889594 34 -0.2862559 0.7764192 0.04903343
# z.var1        0.7718147   1.1347170 34  0.6801826 0.5009945 0.11586471
# z.var2       -0.1116209   0.5268130 34 -0.2118795 0.8334661 0.03631307
#
# ---
#   Residual standard error: 5.11849 on 34 degrees of freedom
# Multiple R-squared: 0.0554183804, Adjusted R-squared: -0.0834906813
# F-statistic: 0.39895 on 5 and 34 DF, p-value: 0.84607

model.output2$statistics
#                Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept) 135.4109966 134.1478457 34  1.0094161 0.3199052 0.17057636
# BLUP.x.var1  -2.1984308   2.2203278 34 -0.9901379 0.3291012 0.16741080
# BLUP.x.var2  -0.6369600   0.8619558 34 -0.7389706 0.4649961 0.12572678
# BLUP.x.var3  -0.5121002   1.7889594 34 -0.2862559 0.7764192 0.04903343
# z.var1        0.7718147   1.1347170 34  0.6801826 0.5009945 0.11586471
# z.var2       -0.1116209   0.5268130 34 -0.2118795 0.8334661 0.03631307
model.output2$rsquared
# 0.0554183804
model.output2$rsquared.adjusted
# -0.0834906813

######## EXAMPLE 3 (after EXAMPLE 2). ADDING A MICRO-MICRO INTERACTION TERM ########
model.formula3 = as.formula(y ~ BLUP.x.var1 * BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2)
model.output3 = micromacro.lm(model.formula3, results$adjusted.group.means, y,
                              results$unequal.groups)
micromacro.summary(model.output3)
# Call:
#   micromacro.lm( y ~ BLUP.x.var1 * BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2, ...)
#
# Residuals:
#        Min        1Q    Median       3Q      Max
# -13.21948 -2.048324 0.7062639 3.843816 7.924922
#
#
# Coefficients:
#                              Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept)             -1.098875e+03 1962.9182021 33 -0.5598169 0.5793848 0.09699214
# BLUP.x.var1              2.231877e+01   38.9620284 33  0.5728339 0.5706400 0.09922547
# BLUP.x.var2              5.988568e+01   96.0256433 33  0.6236426 0.5371496 0.10792809
# BLUP.x.var3             -9.557605e-01    1.9374178 33 -0.4933167 0.6250560 0.08556050
# z.var1                   6.116347e-01    1.1727757 33  0.5215274 0.6054822 0.09041443
# z.var2                  -8.556163e-02    0.5331509 33 -0.1604829 0.8734790 0.02792560
# BLUP.x.var1:BLUP.x.var2 -1.209354e+00    1.9186909 33 -0.6303016 0.5328380 0.10906688
#
# ---
```

```
#   Residual standard error: 5.08795 on 33 degrees of freedom
# Multiple R-squared: 0.0666547309, Adjusted R-squared: -0.103044409
# F-statistic: 0.39278 on 6 and 33 DF, p-value: 0.87831

model.output3$statistics
#                              Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept)             -1.098875e+03 1962.9182021 33 -0.5598169 0.5793848 0.09699214
# BLUP.x.var1              2.231877e+01   38.9620284 33  0.5728339 0.5706400 0.09922547
# BLUP.x.var2              5.988568e+01   96.0256433 33  0.6236426 0.5371496 0.10792809
# BLUP.x.var3             -9.557605e-01    1.9374178 33 -0.4933167 0.6250560 0.08556050
# z.var1                   6.116347e-01    1.1727757 33  0.5215274 0.6054822 0.09041443
# z.var2                  -8.556163e-02    0.5331509 33 -0.1604829 0.8734790 0.02792560
# BLUP.x.var1:BLUP.x.var2 -1.209354e+00    1.9186909 33 -0.6303016 0.5328380 0.10906688
model.output3$rsquared
# 0.0666547309
model.output3$rsquared.adjusted
# -0.103044409

######## EXAMPLE 4 (after EXAMPLE 2). ADDING A MICRO-MACRO INTERACTION TERM ########
model.formula4 = as.formula(y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 * z.var1 + z.var2)
model.output4 = micromacro.lm(model.formula4, results$adjusted.group.means, y,
                              results$unequal.groups)
micromacro.summary(model.output4)
# Call:
#   micromacro.lm( y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 * z.var1 + z.var2, ...)
#
# Residuals:
#        Min        1Q    Median       3Q     Max
# -12.99937 -1.909645 0.8775397 3.712013 8.46591
#
#
# Coefficients:
#                       Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept)        129.22731579 146.4817031 33  0.8822079 0.3840456 0.15179313
# BLUP.x.var1         -2.10556192   2.3951160 33 -0.8791064 0.3857003 0.15127172
# BLUP.x.var2         -0.63762927   0.8747645 33 -0.7289153 0.4711953 0.12587857
# BLUP.x.var3         -0.53590189   1.8273917 33 -0.2932605 0.7711594 0.05098372
# z.var1               2.95426548  19.1170600 33  0.1545356 0.8781288 0.02689146
# z.var2              -0.09852267   0.5467583 33 -0.1801942 0.8581021 0.03135236
# BLUP.x.var3:z.var1   0.21489002   1.8788995 33  0.1143702 0.9096374 0.01990534
#
# ---
#   Residual standard error: 5.11747 on 33 degrees of freedom
# Multiple R-squared: 0.0557926451, Adjusted R-squared: -0.1158814195
# F-statistic: 0.32499 on 6 and 33 DF, p-value: 0.91909

model.output4$statistics
#                       Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept)        129.22731579 146.4817031 33  0.8822079 0.3840456 0.15179313
# BLUP.x.var1         -2.10556192   2.3951160 33 -0.8791064 0.3857003 0.15127172
# BLUP.x.var2         -0.63762927   0.8747645 33 -0.7289153 0.4711953 0.12587857
# BLUP.x.var3         -0.53590189   1.8273917 33 -0.2932605 0.7711594 0.05098372
# z.var1               2.95426548  19.1170600 33  0.1545356 0.8781288 0.02689146
```

```
# z.var2              -0.09852267   0.5467583 33 -0.1801942 0.8581021 0.03135236
# BLUP.x.var3:z.var1   0.21489002   1.8788995 33  0.1143702 0.9096374 0.01990534
model.output4$rsquared
# 0.0557926451
model.output4$rsquared.adjusted
# -0.1158814195
```

`micromacro.summary` *Summarizing the Micro-Macro Multilevel Linear Model Fitting Results*

Description

After fitting a micro-macro multilevel model, this function produces a user-friendly summary table of the results.

Usage

```
micromacro.summary(model.output)
```

Arguments

`model.output`	the output of `micromacro.lm` which contains model results and model specifications.

Details

To date, most multilevel methodologies can only unbiasedly model macro-micro multilevel situations, wherein group-level predictors (e.g., city temperature) are used to predict an individual-level outcome variable (e.g., citizen personality). In contrast, this R package enables researchers to model micro-macro situations, wherein individual-level (micro) predictors (and other group-level predictors) are used to predict a group-level (macro) outcome variable in an unbiased way.

To conduct micro-macro multilevel modeling with the current package, one must first compute the adjusted group means with the function `adjusted.predictors`. This is because in micro-macro multilevel modeling, it is statistically biased to directly regress the group-level outcome variable on the unadjusted group means of individual-level predictors (Croon & van Veldhoven, 2007). Instead, one should use the best linear unbiased predictors (BLUP) of the group means (i.e., the adjusted group means), which is conveniently computed by `adjusted.predictors`.

Once produced by `adjusted.predictors`, the adjusted group means can be used as one of the inputs of the `micromacro.lm` function, which reports estimation results and inferential statistics of the micro-macro multilevel model of interest.

If group size is the same across all groups (i.e., unequal.groups = FALSE), then OLS standard errors are reported and used to determine the inferential statistics in this micro-macro model. If group size is different across groups (i.e., unequal.groups = TRUE), however, then the heteroscedasticity-consistent standard errors are reported and used determine the inferential statistics in this micro-macro model (White, 1980).

`micromacro.summary` produces a detailed summary on the model fitting and specifications, given the outputs of `micromacro.lm`.

Value

table a summary table.

Author(s)

Jackson G. Lu, Elizabeth Page-Gould, Nancy R. Xu (maintainer, nancyranxu@gmail.com).

References

Akinola, M., Page-Gould, E., Mehta, P. H., & Lu, J. G. (2016). Collective hormonal profiles predict group performance. Proceedings of the National Academy of Sciences, 113 (35), 9774-9779.

Croon, M. A., & van Veldhoven, M. J. (2007). Predicting group-level outcome variables from variables measured at the individual level: a latent variable multilevel model. Psychological methods, 12(1), 45-57.

See Also

micromacro.lm for fitting the micro-macro multilevel linear model of interest.

Examples

```
######## SETUP: DATA GENERATING PROCESSES ########
set.seed(123)
# Step 1. Generate a G-by-q data frame of group-level predictors (e.g., control variables), z.data
# In this example, G = 40, q = 2
group.id = seq(1, 40)
z.var1 = rnorm(40, mean=0, sd=1)
z.var2 = rnorm(40, mean=100, sd=2)
z.data = data.frame(group.id, z.var1, z.var2)
# Step 2. Generate a G-by-p data frame of group-level means for the predictors that will be used to
# generate x.data
# In this example, there are 3 individual-level predictors, thus p = 3
x.var1.means = rnorm(40, mean=50, sd = .05)
x.var2.means = rnorm(40, mean=20, sd = .05)
x.var3.means = rnorm(40, mean=-10, sd = .05)
x.data.means = data.frame(group.id, x.var1.means, x.var2.means, x.var3.means)
# Step 3. Generate two N-by-p data frames of individual-level predictors, x.data
# One of these two data frames assumes unequal-sized groups (Step 3a), whereas the other assumes
# equal-sized groups (Step 3b):
# Step 3a. Generate the individual-level predictors
# In this example, N = 200 and group size is unequal
x.data.unequal = data.frame( group.id=rep(1:40, times=sample( c(4,5,6), 40, replace=TRUE) )[1:200] )
x.data.unequal = merge( x.data.unequal,
           data.frame( group.id, x.var1.means, x.var2.means, x.var3.means ), by="group.id" )
x.data.unequal = within( x.data.unequal, {
  x.var1 = x.var1.means + rnorm(200, mean=0, sd = 2)
  x.var2 = x.var2.means + rnorm(200, mean=0, sd = 6)
  x.var3 = x.var3.means + rnorm(200, mean=0, sd = 1.5)
})
# Step 3b. Generate the individual-level predictors
# In this example, N = 200 and group size is equal
```

```
x.data.equal = data.frame( group.id=rep(1:40, each=5) )
x.data.equal = merge( x.data.equal, x.data.means, by="group.id" )
x.data.equal = within( x.data.equal, {
  x.var1 = x.var1.means + rnorm(200, mean=0, sd = 2)
  x.var2 = x.var2.means + rnorm(200, mean=0, sd = 6)
  x.var3 = x.var3.means + rnorm(200, mean=0, sd = 1.5)
})
# Step 3. Generate a G-by-1 data frame of group-level outcome variable, y
# In this example, G = 40
y = rnorm(40, mean=6, sd=5)

apply(x.data.equal,2,mean)
#    group.id x.var1.means x.var2.means x.var3.means      x.var3      x.var2      x.var1
# 20.500000    50.000393    19.994708    -9.999167   -10.031995    20.185361    50.084635
apply(x.data.unequal,2,mean)
#    group.id x.var1.means x.var2.means x.var3.means      x.var3      x.var2      x.var1
# 20.460000    50.002286    19.994605    -9.997034    -9.983146    19.986111    50.123591
apply(z.data,2,mean)
# z.var1      z.var2
# 0.04518332 99.98656817
mean(y)
# 6.457797

######## EXAMPLE 1. GROUP SIZE IS DIFFERENT ACROSS GROUPS ########
######## Need to use adjusted.predictors() in the same package ###

# Step 4a. Generate a G-by-1 matrix of group ID, z.gid. Then generate an N-by-1 matrix of
# each individual's group ID, x.gid, where the group sizes are different
z.gid = seq(1:40)
x.gid = x.data.unequal$group.id
# Step 5a. Generate the best linear unbiased predictors that are calcualted from
# individual-level data
x.data = x.data.unequal[,c("x.var1","x.var2","x.var3")]
results = adjusted.predictors(x.data, z.data, x.gid, z.gid)
# Note: Given the fixed random seed, the output should be as below
results$unequal.groups
# TRUE
names(results$adjusted.group.means)
# "BLUP.x.var1" "BLUP.x.var2" "BLUP.x.var3" "z.var1"      "z.var2"      "gid"
head(results$adjusted.group.means)
#   BLUP.x.var1 BLUP.x.var2 BLUP.x.var3 group.id      z.var1    z.var2 gid
# 1    50.05308    20.83911  -10.700361        1 -0.56047565  98.61059   1
# 2    48.85559    22.97411   -9.957270        2 -0.23017749  99.58417   2
# 3    50.16357    19.50001   -9.645735        3  1.55870831  97.46921   3
# 4    49.61853    21.25962  -10.459398        4  0.07050839 104.33791   4
# 5    50.49673    21.38353   -9.789924        5  0.12928774 102.41592   5
# 6    50.86154    19.15901   -9.245675        6  1.71506499  97.75378   6
# Step 6a. Fit a micro-macro multilevel model when group sizes are different
model.formula = as.formula(y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2)
model.output = micromacro.lm(model.formula, results$adjusted.group.means, y, results$unequal.groups)
micromacro.summary(model.output)
# Call:
#   micromacro.lm( y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2, ...)
```

```
#
# Residuals:
#        Min        1Q  Median       3Q      Max
# -13.41505 -2.974074 1.13077 3.566021 6.975819
#
#
# Coefficients:
#           Estimate  Uncorrected S.E. Corrected S.E. df          t  Pr(>|t|)          r
# (Intercept) 78.1232185    121.5103390  122.1367432 34  0.6396373 0.5266952 0.10904278
# BLUP.x.var1 -0.7589602      1.4954434    1.7177575 34 -0.4418320 0.6614084 0.07555696
# BLUP.x.var2  0.4263309      0.7070773    0.6299759 34  0.6767416 0.5031484 0.11528637
# BLUP.x.var3  0.2658078      2.4662049    2.4051691 34  0.1105152 0.9126506 0.01894980
# z.var1       0.4315941      1.0855707    1.0614535 34  0.4066068 0.6868451 0.06956356
# z.var2      -0.3949955      0.5573789    0.4230256 34 -0.9337390 0.3570228 0.15812040
#
# ---
#   Residual standard error: 5.1599 on 34 degrees of freedom
# Multiple R-squared: 0.0400727607, Adjusted R-squared: -0.1010930098
# F-statistic: 0.28387 on 5 and 34 DF, p-value: 0.91869

model.output$statistics
#           Estimate  Uncorrected S.E. Corrected S.E. df          t  Pr(>|t|)          r
# (Intercept) 78.1232185    121.5103390  122.1367432 34  0.6396373 0.5266952 0.10904278
# BLUP.x.var1 -0.7589602      1.4954434    1.7177575 34 -0.4418320 0.6614084 0.07555696
# BLUP.x.var2  0.4263309      0.7070773    0.6299759 34  0.6767416 0.5031484 0.11528637
# BLUP.x.var3  0.2658078      2.4662049    2.4051691 34  0.1105152 0.9126506 0.01894980
# z.var1       0.4315941      1.0855707    1.0614535 34  0.4066068 0.6868451 0.06956356
# z.var2      -0.3949955      0.5573789    0.4230256 34 -0.9337390 0.3570228 0.15812040
model.output$rsquared
# 0.0400727607
model.output$rsquared.adjusted
# -0.1010930098

######## EXAMPLE 2. GROUP SIZE IS THE SAME ACROSS ALL GROUPS ########
######## Need to use adjusted.predictors() in the same package ######

# Step 4b. Generate a G-by-1 matrix of group ID, z.gid. Then generate an N-by-1 matrix of
# each individual's group ID, x.gid, where group size is the same across all groups
z.gid = seq(1:40)
x.gid = x.data.equal$group.id
# Step 5b. Generate the best linear unbiased predictors that are calcualted from
# individual-level data
x.data = x.data.equal[,c("x.var1","x.var2","x.var3")]
results = adjusted.predictors(x.data, z.data, x.gid, z.gid)
results$unequal.groups
# FALSE
names(results$adjusted.group.means)
# "BLUP.x.var1" "BLUP.x.var2" "BLUP.x.var3" "z.var1"      "z.var2"      "gid"
results$adjusted.group.means[1:5, ]
#   BLUP.x.var1 BLUP.x.var2 BLUP.x.var3 group.id      z.var1    z.var2 gid
# 1    50.91373    19.12994  -10.051647        1 -0.56047565  98.61059   1
# 2    50.19068    19.17978  -10.814382        2 -0.23017749  99.58417   2
# 3    50.13390    20.98893   -9.952348        3  1.55870831  97.46921   3
```

```
# 4    49.68169    19.60632  -10.612717        4  0.07050839 104.33791   4
# 5    50.28579    22.07469  -10.245505        5  0.12928774 102.41592   5
# Step 6b. Fit a micro-macro multilevel model when group size is the same across groups
model.output2 = micromacro.lm(model.formula, results$adjusted.group.means, y,
                              results$unequal.groups)
micromacro.summary(model.output2)
# Call:
#   micromacro.lm( y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2, ...)
#
# Residuals:
#        Min        1Q    Median      3Q      Max
# -12.94409 -1.898937 0.8615494 3.78739 8.444582
#
#
# Coefficients:
#                Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept) 135.4109966 134.1478457 34  1.0094161 0.3199052 0.17057636
# BLUP.x.var1  -2.1984308   2.2203278 34 -0.9901379 0.3291012 0.16741080
# BLUP.x.var2  -0.6369600   0.8619558 34 -0.7389706 0.4649961 0.12572678
# BLUP.x.var3  -0.5121002   1.7889594 34 -0.2862559 0.7764192 0.04903343
# z.var1        0.7718147   1.1347170 34  0.6801826 0.5009945 0.11586471
# z.var2       -0.1116209   0.5268130 34 -0.2118795 0.8334661 0.03631307
#
# ---
#   Residual standard error: 5.11849 on 34 degrees of freedom
# Multiple R-squared: 0.0554183804, Adjusted R-squared: -0.0834906813
# F-statistic: 0.39895 on 5 and 34 DF, p-value: 0.84607

model.output2$statistics
#                Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept) 135.4109966 134.1478457 34  1.0094161 0.3199052 0.17057636
# BLUP.x.var1  -2.1984308   2.2203278 34 -0.9901379 0.3291012 0.16741080
# BLUP.x.var2  -0.6369600   0.8619558 34 -0.7389706 0.4649961 0.12572678
# BLUP.x.var3  -0.5121002   1.7889594 34 -0.2862559 0.7764192 0.04903343
# z.var1        0.7718147   1.1347170 34  0.6801826 0.5009945 0.11586471
# z.var2       -0.1116209   0.5268130 34 -0.2118795 0.8334661 0.03631307
model.output2$rsquared
# 0.0554183804
model.output2$rsquared.adjusted
# -0.0834906813

######## EXAMPLE 3 (after EXAMPLE 2). ADDING A MICRO-MICRO INTERACTION TERM ########
model.formula3 = as.formula(y ~ BLUP.x.var1 * BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2)
model.output3 = micromacro.lm(model.formula3, results$adjusted.group.means, y,
                              results$unequal.groups)
micromacro.summary(model.output3)
# Call:
#   micromacro.lm( y ~ BLUP.x.var1 * BLUP.x.var2 + BLUP.x.var3 + z.var1 + z.var2, ...)
#
# Residuals:
#        Min        1Q    Median       3Q      Max
# -13.21948 -2.048324 0.7062639 3.843816 7.924922
#
```

```
#
# Coefficients:
#                               Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept)              -1.098875e+03 1962.9182021 33 -0.5598169 0.5793848 0.09699214
# BLUP.x.var1               2.231877e+01   38.9620284 33  0.5728339 0.5706400 0.09922547
# BLUP.x.var2               5.988568e+01   96.0256433 33  0.6236426 0.5371496 0.10792809
# BLUP.x.var3              -9.557605e-01    1.9374178 33 -0.4933167 0.6250560 0.08556050
# z.var1                    6.116347e-01    1.1727757 33  0.5215274 0.6054822 0.09041443
# z.var2                   -8.556163e-02    0.5331509 33 -0.1604829 0.8734790 0.02792560
# BLUP.x.var1:BLUP.x.var2 -1.209354e+00    1.9186909 33 -0.6303016 0.5328380 0.10906688
#
# ---
#   Residual standard error: 5.08795 on 33 degrees of freedom
# Multiple R-squared: 0.0666547309, Adjusted R-squared: -0.103044409
# F-statistic: 0.39278 on 6 and 33 DF, p-value: 0.87831

model.output3$statistics
#                               Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept)              -1.098875e+03 1962.9182021 33 -0.5598169 0.5793848 0.09699214
# BLUP.x.var1               2.231877e+01   38.9620284 33  0.5728339 0.5706400 0.09922547
# BLUP.x.var2               5.988568e+01   96.0256433 33  0.6236426 0.5371496 0.10792809
# BLUP.x.var3              -9.557605e-01    1.9374178 33 -0.4933167 0.6250560 0.08556050
# z.var1                    6.116347e-01    1.1727757 33  0.5215274 0.6054822 0.09041443
# z.var2                   -8.556163e-02    0.5331509 33 -0.1604829 0.8734790 0.02792560
# BLUP.x.var1:BLUP.x.var2 -1.209354e+00    1.9186909 33 -0.6303016 0.5328380 0.10906688
model.output3$rsquared
# 0.0666547309
model.output3$rsquared.adjusted
# -0.103044409

######## EXAMPLE 4 (after EXAMPLE 2). ADDING A MICRO-MACRO INTERACTION TERM ########
model.formula4 = as.formula(y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 * z.var1 + z.var2)
model.output4 = micromacro.lm(model.formula4, results$adjusted.group.means, y,
                              results$unequal.groups)
micromacro.summary(model.output4)
# Call:
#   micromacro.lm( y ~ BLUP.x.var1 + BLUP.x.var2 + BLUP.x.var3 * z.var1 + z.var2, ...)
#
# Residuals:
#        Min        1Q    Median       3Q     Max
# -12.99937 -1.909645 0.8775397 3.712013 8.46591
#
#
# Coefficients:
#                        Estimate         S.E. df          t  Pr(>|t|)          r
# (Intercept)        129.22731579 146.4817031 33  0.8822079 0.3840456 0.15179313
# BLUP.x.var1         -2.10556192   2.3951160 33 -0.8791064 0.3857003 0.15127172
# BLUP.x.var2         -0.63762927   0.8747645 33 -0.7289153 0.4711953 0.12587857
# BLUP.x.var3         -0.53590189   1.8273917 33 -0.2932605 0.7711594 0.05098372
# z.var1               2.95426548  19.1170600 33  0.1545356 0.8781288 0.02689146
# z.var2              -0.09852267   0.5467583 33 -0.1801942 0.8581021 0.03135236
# BLUP.x.var3:z.var1   0.21489002   1.8788995 33  0.1143702 0.9096374 0.01990534
#
```

```
# ---
#   Residual standard error: 5.11747 on 33 degrees of freedom
# Multiple R-squared: 0.0557926451, Adjusted R-squared: -0.1158814195
# F-statistic: 0.32499 on 6 and 33 DF, p-value: 0.91909

model.output4$statistics
#                       Estimate        S.E. df          t  Pr(>|t|)          r
# (Intercept)        129.22731579 146.4817031 33  0.8822079 0.3840456 0.15179313
# BLUP.x.var1         -2.10556192   2.3951160 33 -0.8791064 0.3857003 0.15127172
# BLUP.x.var2         -0.63762927   0.8747645 33 -0.7289153 0.4711953 0.12587857
# BLUP.x.var3         -0.53590189   1.8273917 33 -0.2932605 0.7711594 0.05098372
# z.var1               2.95426548  19.1170600 33  0.1545356 0.8781288 0.02689146
# z.var2              -0.09852267   0.5467583 33 -0.1801942 0.8581021 0.03135236
# BLUP.x.var3:z.var1   0.21489002   1.8788995 33  0.1143702 0.9096374 0.01990534
model.output4$rsquared
# 0.0557926451
model.output4$rsquared.adjusted
# -0.1158814195
```

Index