# Package 'Dire'

April 3, 2021

**Version** 1.0.3

**Date** 2021-03-17

**Title** Linear Regressions with a Latent Outcome Variable

**Maintainer** Paul Bailey <pbailey@air.org>

**Depends** R (>= 3.3.0)

**Imports** minqa, foreach, iterators, methods, Matrix, haven, Rcpp

**Description** Fit linear models, estimating score distributions for groups of people, following Cohen and Jiang (1999) <doi:10.2307/2669917>. In this model, the response is a latent trait (such as student ability) and raw item responses are combined with item difficulties in an item response theory (IRT) framework to form a density for each unit (student). This latent trait is then integrated out. This software is intended to fit the same models as the existing software 'AM' <http://am.air.org/>.

**License** GPL-2

**VignetteBuilder** knitr

**Suggests** knitr, rmarkdown, testthat, withr, doParallel, EdSurvey

**URL** https://american-institutes-for-research.github.io/Dire/

**BugReports** https://github.com/American-Institutes-for-Research/Dire/issues

**ByteCompile** true

**Note** This publication was prepared for NCES under Contract No. ED-IES-12-D-0002 with the American Institutes for Research. Mention of trade names, commercial products, or organizations does not imply endorsement by the U.S. Government.

**Encoding** UTF-8

**RoxygenNote** 7.1.1

**LinkingTo** Rcpp

**NeedsCompilation** yes

**Author** Harold Doran [aut],
Paul Bailey [aut, cre],
Eric Buehler [aut],
Sun-joo Lee [aut],

Claire Kelley [ctb],
Emmanuel Sikali [pdr]

**Repository** CRAN

**Date/Publication** 2021-04-03 17:10:02 UTC

# R topics documented:



---

`mml` *Marginal Maximum Likelihood Estimation of Linear Models*

---

## Description

Implements a survey-weighted marginal maximum estimation, a type of regression where the outcome is a latent trait (such as student ability). Instead of using an estimate, the likelihood function marginalizes student ability. Includes a variety of variance estimation strategies.

## Usage

```
mml(
  formula,
  stuItems,
  stuDat,
  idVar,
  dichotParamTab = NULL,
  polyParamTab = NULL,
  testScale = NULL,
  Q = 30,
  minNode = -4,
  maxNode = 4,
  polyModel = c("GPCM", "GRM"),
  weightVar = NULL,
  multiCore = FALSE,
  bobyqaControl = list(maxfun = 1e+05),
  composite = TRUE,
  strataVar = NULL,
  PSUVar = NULL,
  fast = TRUE
)
```

## Arguments

| | |
|---|---|
| formula | a formula object in the style of lm |
| stuItems | a list where each element is named a student ID and contains a data.frame; see Details for the format |
| stuDat | a data.frame with a single row per student. Predictors in the formula must be in stuDat. |
| idVar | a variable name on stuDat that is the identifier. Every ID from stuDat must appear on stuItems and vice versa. |
| dichotParamTab | a data.frame of dichotimous item information, see Details |
| polyParamTab | a data.frame of polytimous item information, see Details |
| testScale | a data.frame of scaling information, see Details |
| Q | an integer; the number of integration points |
| minNode | a numeric; the smallest integration point for the latent variable |
| maxNode | a numeric; the largest integration point for the latent variable |
| polyModel | polytomous response model; one of GPCM for the Graded Partial Credit Model or GRM for the Graded Response Model |
| weightVar | a variable name on stuDat that is the full sample weight |
| multiCore | allows the foreach package to be used. You should have already setup the the registerDoParallel function in the doParallel package. |
| bobyqaControl | a list that gets passed to the bobyqa optimizer in minqa |
| composite | a logical indicating if an overall test should be treated as a composite score; a composite is a weighted average of the subscales in it. |
| strataVar | character naming a variable on stuDat, the variable indicating the stratum for each row. Used in post-hoc robust variance estimation. |
| PSUVar | character naming a variable on stuDat; the primary sampling unit (PSU) variable. Used in post-hoc robust variance estimation. The values do not need to be unique across strata. |
| fast | a logical indicating if cpp code should be used in mml processes. This should yield speed-ups to runs. |

## Details

The mml function models a latent outcome conditioning on item response data, covariate data, item parameter information, and scaling information. These four parts are broken up into at least one argument each. Student item response data go into stuItems; whereas student covariates, weights, and sampling information go into stuDat. The dichotParamTab and polyParamTab contains item parameter information for dichotimous and polytimous items, respectively—the item parameter data is the result of an existing item parameter scaling. In the case of the National Assessment of Educational Progress (NAEP), they can be found online, for example, at https://nces.ed.gov/nationsreportcard/tdw/analysis/scali
Finally, information about scaling and subscale weights for composites are put in testScale.

The model for dichotomous responses data is, by default, three Parameter Logit (3PL), unless the item parameter information provided by users suggests otherwise. For example, if the scaling used

a two Parameter Logit (2PL) model, then the guessing parameter can simply be set to zero. For polytomous responses data, the model is dictated by the polyModel argument.

The dichotParamTab argument is a data.frame with a column named ItemID that identifies the items and agrees with the key column in the stuItems argument, and, for a 3PL item, columns slope, difficulty, and guessing for the "a", "d", and "g" parameters, respectively; see the vignette for details of the 3PL model. Users can also use the column names directly from the vignette notation ("a", "d", and "g") if they prefer. Items that are missing (NA) are not used in the likelihood function Users wishing to apply a special behavior for a subset of items can use set those items to an invalid score and put that in the dichotParamTab column missingCode. They are then scored as if they are missingValue proportion correct. To use the guessing parameter for the proportion correct set missingValue to "c".

The polyParamTab has columns ItemID that must match with the key from stuItems, as well as slope (which can also be called a) that corresponds to the a parameter in the vignette. Users must also specify the location of the cut points (d-sub-cj in the vignette) which are named d1, d2, ..., up to dn where n is one less than the number of score points. Some people prefer to also apply a shift to all of these and this shift is applied when there is a column named itemLocation by simply adding that to every d* column. Items are not included in the likelihood for an individual when there value on stuItems is NA, but no provision is made for guessing, nor special provision for missing codes in polytimious items.

For both dichotParamTab and polyParamTab users wishing to use a D paramter of 1.7 (or any other value) may specify that, per item, in a column named D.

When there are multiple constructs, subscales, or the user wants a composite score, additional, optional, columns test and subtest can be used. these columns can be numeric or text, they must agree with the same columns in testScale to scale the results.

Student data are broken up into two parts. The item response data goes into stuItems, and the student covariates for the formula go into stuDat. Information about items, such as item difficulties, is in paramTab. All dichotomous items are assumed to be 3PL, though by setting the guessing parameter to zero, the user can use a 2PL or the one Parameter Logit (1PL) or Rasch models. The model for polytomous responses data is dictated by the polyModel argument.

The marginal maximum likelihood then integrates the product of the student ability from the assessment data, and the estimate from the linear model estimates each student's ability based on the formula provided and a residual standard error term. This integration happens from the minimum node to the maximum node in the control argument (described later in this section) with Q quadrature points.

The stuItems argument has the scored student data. It is a list where each element is named with student ID and contains a data.frame with at least two columns. The first required column is named key and shows the item name as it appears in paramTab; the second column in named score and shows the score for that item. For binomial items, the score is 0 or 1. For GPCM items, the scores start at zero as well. For GRM, the scores start at 1.

For a GPCM model, P0 is the "a" parameter, and the other columns are the "d" parameters; see the vignette for details of the GPCM model.

The quadrature points then are a range from minNode to maxNode with a total of Q nodes.

## Value

when called for a single subscale or overall score, returns object of class `mmlMeans`. This is a list with elements:

- `call` the call used to generate this `mml.means` object
- `coefficients` the unscaled marginal maximum likelihood regression coefficients
- `LogLik` the log-likelihood of the fit model
- `X` the design matrix of the marginal maximum likelihood regression
- `Convergence` a convergence note from the `bobyqa` optimizer
- `location` used for scaling the estimates
- `scale` used for scaling the estimates
- `lnlf` the log-likelihood function of the unscaled parameters
- `rr1` the density function of each individual, conditional only on item responses in `stuItems`
- `stuDat` the `stuDat` argument
- `weightVar` the name of the weight variable on `stuDat`
- `nodes` the nodes the likelihood was evaluated on
- `iterations` the number of iterations required to reach convergence
- `obs` the number of observations used
- `weightedObs` the weighted N for the observations
- `strataVar` the column name of the stratum variable on stuDat; potentially used for variance estimation
- `PSUVar` the column name of the stratum variable on stuDat; potentially used for variance estimation

When a composite score is computed there are several subscales run and the return is a `mmlCompositeMeans`. Many elements are themselves list with one element per construct. this is a list with elements:

- `call` the call used to generate this `mml.means` object
- `coefficients` matrix of the unscaled marginal maximum likelihood regression coefficients, each row represents a subscale, each column represents a coefficient
- `X` the design matrix of the marginal maximum likelihood regression
- `rr1` a list of elements, each the rr1 object for a subscale (see `mmlMeans` output)
- `ids` The ID variable used for each row of `stuDat`
- `Convergence` a vector of convergence notes from the `bobyqa` optimizer
- `lnlfl` a list of log-likelihood functions of the unscaled parameters, by construct
- `stuDat` a list of `stuDat` data frames, as used when fitting each construct, filtered to just relevant student records
- `weightVar` the name of the weight variable on `stuDat`
- `nodes` the nodes the likelihood was evaluated on
- `iterations` a vector of the number of iterations required to reach convergence on each construct

- obs a vector of the the number of observations used on each construct
- testScale the testScale used to scale the data
- weightedObs a vector of the weighted N for the observations
- SubscaleVC the covariance matrix of subscales. The residuals are assumed to be multivariate normal with this covairiance matrix
- idVar the name of the identifier used on stuDat and stuItems data
- resl list of mmlMeans objects, one per construct
- strataVar the column name of the stratum variable on stuDat; potentially used for variance estimation
- PSUVar the column name of the stratum variable on stuDat; potentially used for variance estimation

LogLik is not returned because there is no likelihood for a composite model.

## Author(s)

Harold Doran, Paul Bailey, Claire Kelley, Sun-joo Lee, and Eric Buehler

## Examples

```
## Not run:
require(EdSurvey)

# 1) make param tab for dichotomous items
dichotParamTab <- data.frame(ItemID = c("m109801", "m020001", "m111001",
                                        "m046301", "m046501", "m051501",
                                        "m111601", "m111301", "m111201",
                                        "m110801", "m110101"),
                             test = rep("composite",11),
                             subtest = c(rep("num",6),rep("alg",5)),
                             slope = c(0.96, 0.69, 0.83,
                                       0.99, 1.03, 0.97,
                                       1.45, 0.59, 0.34,
                                       0.18, 1.20),
                             difficulty = c(-0.37, -0.55,  0.85,
                                            -0.97, -0.14,  1.21,
                                             0.53, -1.84, -0.46,
                                             2.43,  0.70),
                             guessing = c(0.16, 0.00, 0.17,
                                          0.31, 0.37, 0.18,
                                          0.28, 0.15, 0.09,
                                          0.05, 0.18),
                             D = rep(1.7, 11),
                             MODEL = rep("3pl", 11))

# param tab for GPCM items
polyParamTab <- data.frame(ItemID = factor(c("m0757cl", "m066501")),
                           test = rep("composite",2),
                           subtest = rep("alg",2),
                           slope = c(0.43, 0.52), # could also be called "a"
```

```
                            itemLocation = c(-1.21, -0.96), # added to d1 ... dn
                            d1 = c(2.38, -0.56),
                            d2 = c(-0.57, 0.56),
                            d3 = c(-1.18, NA),
                            D = c(1.7, 1.7),
                          scorePoints = c(4L, 3L)) # number of score points, read d1 to d(n-1)
# read-in NAEP Primer data
sdf <- readNAEP(system.file("extdata/data", "M36NT2PM.dat", package = "NAEPprimer"))
# read in these items
items <- c(as.character(dichotParamTab$ItemID), as.character(polyParamTab$ItemID))
# dsex, student sex
# origwt, full sample weights
# repgrp1, stratum indicator
# jkunit, PSU indicator
edf <- getData(data=sdf, varnames=c(items, "dsex", "origwt", "repgrp1", "jkunit", "sdracem"),
               omittedLevels = FALSE, returnJKreplicates=FALSE)
# make up a student ID
edf$sid <- paste0("S",1:nrow(edf))
# apply simplified scoring
for(i in 1:length(items)) {
  coli <- items[i]
  # save the original
  rawcol <- paste0(coli,"raw")
  edf[,rawcol] <- edf[,coli]
  if( coli %in% dichotParamTab$ItemID) {
    edf[,coli] <- ifelse(grepl("[ABCDE]", edf[,rawcol]), 0, NA)
    edf[,coli] <- ifelse(grepl("*", edf[,rawcol]), 1, edf[,coli])
  } else {
    # scale for m066501
    edf[,coli] <- ifelse(grepl("Incorrect", edf[,rawcol]), 0, NA)
    edf[,coli] <- ifelse(grepl("Partial", edf[,rawcol]), 1, edf[,coli])
    edf[,coli] <- ifelse(grepl("Correct", edf[,rawcol]), 2, edf[,coli])
    # scale for m0757cl
    edf[,coli] <- ifelse(grepl("None correct", edf[,rawcol]), 0, edf[,coli])
    edf[,coli] <- ifelse(grepl("One correct", edf[,rawcol]), 1, edf[,coli])
    edf[,coli] <- ifelse(grepl("Two correct", edf[,rawcol]), 2, edf[,coli])
    edf[,coli] <- ifelse(grepl("Three correct", edf[,rawcol]), 3, edf[,coli])
  }
  edf[,rawcol] <- NULL # delete original
}

# stuItems has one row per student/item combination
stuItems <- edf[,c("sid", items)]
stuItems <- reshape(data=stuItems, varying=c(items), idvar=c("sid"),
                    direction="long", v.names="score", times=items, timevar="key")
# stuDat is one row per student an contains student-level information
stuDat <- edf[,c("sid", "origwt", "repgrp1", "jkunit", "dsex", "sdracem")]

# testDat shows scaling and weights for subtests, an overall score can be treated as a subtest
testDat <- data.frame(test=c("composite", "composite") ,
                      subtest=c("num", "alg"),
                      location=c(277.1563, 280.2948),
                      scale=c(37.7297, 36.3887),
```

```
                    subtestWeight=c(0.3,0.7))

# estimate a regression for Algebra subscale
mmlA <- mml(alg ~ dsex,
            stuItems=stuItems, stuDat=stuDat,
            dichotParamTab=dichotParamTab, polyParamTab=polyParamTab,
            testScale=testDat,
            idVar="sid", weightVar="origwt", # these are column names on stuDat
            strataVar="repgrp1", PSUVar="jkunit")
# summary, with Taylor standard errors
summary(mmlA, varType="Taylor")

# composite regression
mmlC <- mml(composite ~ dsex ,
            stuItems=stuItems, stuDat=stuDat,
            dichotParamTab=dichotParamTab, polyParamTab=polyParamTab,
            testScale=testDat,
            idVar="sid", weightVar="origwt", # these are column names on stuDat
            strataVar="repgrp1", PSUVar="jkunit")
# summary, with Taylor standard errors
summary(mmlC, varType="Taylor")


## End(Not run)
```

# Index