

Package: Dire (via r-universe)

October 13, 2026

Version 3.0.0

Date 2025-02-28

Title Linear Regressions with a Latent Outcome Variable

Maintainer Paul Bailey <pbailey@air.org>

Depends R (>= 4.0.0)

Imports methods, Matrix, haven, Rcpp (>= 1.0.8.3), lbfgs, MASS, data.table, tidyr, dplyr, cli, lifecycle, mvnfast

Description Fit latent variable linear models, estimating score distributions for groups of people, following Cohen and Jiang (1999) <doi:10.2307/2669917>. In this model, a latent distribution is conditional on students item response, item characteristics, and conditioning variables the user includes. This latent trait is then integrated out. This software is intended to fit the same models as the existing software 'AM' <<https://am.air.org/>>. As of version 2, also allows the user to draw plausible values.

License GPL-2

VignetteBuilder knitr

Suggests knitr, rmarkdown, testthat, withr, 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-0002/0004 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 7.3.2

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Author Emmanuel Sikali [pdr], Paul Bailey [aut, cre]
 (<<https://orcid.org/0000-0003-0989-8729>>), Eric Buehler [aut],
 Sun-joo Lee [aut], Harold Doran [aut], Blue Webb [aut]
 (<<https://orcid.org/0009-0004-4080-9864>>), Ali Fathi [ctb],
 Claire Kelley [ctb]

Config/pak/sysreqs make libicu-dev libx11-dev zlib1g-dev

Repository <https://american-institutes-for-research.r-universe.dev>

Date/Publication 2025-09-19 03:55:12 UTC

RemoteUrl <https://github.com/american-institutes-for-research/dire>

RemoteRef HEAD

RemoteSha ed3d7ea07f8c701cb9740d5b222283943998c92c

Contents

drawPVs	2
mml	4

Index	12
--------------	-----------

drawPVs	<i>Draw plausible values (PVs) from an mml fit</i>
---------	--

Description

Draw plausible values (PVs) from an mml fit

Usage

```
drawPVs(
  x,
  npv,
  pvVariableNameSuffix = "_dire",
  stochasticBeta = deprecated(),
  ...
)

## S3 method for class 'summary.mmlMeans'
drawPVs(
  x,
  npv = 5L,
  pvVariableNameSuffix = "_dire",
  stochasticBeta = deprecated(),
  ...
)
```

```
## S3 method for class 'mmlMeans'
drawPVs(
  x,
  npv = 5L,
  pvVariableNameSuffix = "_dire",
  stochasticBeta = deprecated(),
  ...
)

## S3 method for class 'summary.mmlCompositeMeans'
drawPVs(
  x,
  npv = 20L,
  pvVariableNameSuffix = "_dire",
  stochasticBeta = deprecated(),
  ...
)

## S3 method for class 'mmlCompositeMeans'
drawPVs(
  x,
  npv = 20L,
  pvVariableNameSuffix = "_dire",
  stochasticBeta = deprecated(),
  ...
)
```

Arguments

<code>x</code>	a fit from a call to mml
<code>npv</code>	integer indicating the number of plausible values to draw
<code>pvVariableNameSuffix</code>	suffix added to new PV variables after construct name and before the plausible value ID. For example, if there is a construct <code>math</code> and the suffix is the default <code>_dire</code> , then the fourth plausible value would have a column name, <code>math_dire4</code> .
<code>stochasticBeta</code>	deprecated.
<code>...</code>	additional parameters passed on to methods.

Details

When the argument passed to `stochasticBeta` is a data frame then each column is an element that will be used as a regression coefficient for that index of the coefficients vector. The row index used for the *n*th PV will be the *n*th row.

Value

the `stuDat` attribute of the [mml](#) fit object, `x`, with columns for the new plausible values.

Author(s)

Paul Bailey, Sun-joo Lee, and Eric Buehler

Examples

```
# See Examples in mml
```

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 = NULL,  
  stuDat,  
  idVar,  
  dichotParamTab = NULL,  
  polyParamTab = NULL,  
  testScale = NULL,  
  Q = 66,  
  minNode = -5,  
  maxNode = 5,  
  polyModel = c("GPCM", "GRM"),  
  weightVar = NULL,  
  multiCore = FALSE,  
  bobyqaControl = NULL,  
  composite = TRUE,  
  strataVar = NULL,  
  PSUVar = NULL,  
  fast = NULL,  
  calcCor = NULL,  
  verbose = 0,  
  retainedInformation = 1,  
  optimizer = c("EM", "QN")  
)
```

Arguments

formula	a formula object in the style of <code>lm</code>
stuItems	deprecated. Simply put items in <code>stuDat</code> now.
stuDat	a <code>data.frame</code> with a single row per student. Predictors in the formula must be in <code>stuDat</code> as well as items.
idVar	a variable name on <code>stuDat</code> that is the identifier. Every ID from <code>stuDat</code> must appear on <code>stuItems</code> and vice versa.
dichotParamTab	a <code>data.frame</code> of dichotomous item information, see Details
polyParamTab	a <code>data.frame</code> of polytomous item information, see Details
testScale	a <code>data.frame</code> 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 <code>stuDat</code> that is the full sample weight
multiCore	a logical indicating whether to use parallel processing
bobyqaControl	deprecated. A list that gets passed to the bobyqa optimizer in <code>minqa</code>
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 <code>stuDat</code> , the variable indicating the stratum for each row. Used in post-hoc robust variance estimation.
PSUVar	character naming a variable on <code>stuDat</code> ; the primary sampling unit (PSU) variable. Used in post-hoc robust variance estimation. The values do not need to be unique across strata.
fast	deprecated. Always TRUE now.
calcCor	deprecated. Always TRUE now.
verbose	integer, negative or zero for no details, increasingly verbose messages at one and two
retainedInformation	set to a value of 1 to fit the model as is typically fit. If the value is less than one, a principal component analysis is performed and columns associated with principal components totaling at least <code>retainedInformation</code> are retained while all other data is discarded. The estimated latent regression coefficients will accordingly be named as PC1, ..., PCN.
optimizer	character naming the optimization method to use; one of EM for Expectation Maximization or QN for Quasi-Newton

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` contain item parameter information for dichotomous and polytomous 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 [NAEP technical documentation](#). 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_{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 their value on `stuItems` is NA, but no provision is made for guessing, nor special provision for missing codes in polytomous items.

For both `dichotParamTab` and `polyParamTab` users wishing to use a D parameter 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 is named `score` and shows the score for that item. For dichotomous 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, P_0 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
- `getX` a function that, when called on `stuDat`, returns the design matrix of the marginal maximum likelihood regression
- `Convergence` a convergence note from the optimizer
- `location` used for scaling the estimates
- `scale` used for scaling the estimates
- `lnlf` the log-likelihood function of the unscaled parameters
- `funcs` a list of functions that may be evaluated on a vector of parameters. The first returns the log-likelihood of the marginal maximum likelihood regression evaluated at the given parameters. The second and third correspond to the first and second derivatives, respectively, of the MML regression evaluated at the parameters.
- `rr1` the density function of each individual, conditional only on item responses in `stuItems`
- `stuDat` the `stuDat` argument
- `likelihood_stus` `rr1`, but pivoted long so that each row corresponds to a student.
- `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
- `testScale` the `testScale` used to scale the data
- `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 PSU variable on `stuDat`; potentially used for variance estimation
- `itemScorePoints` a data frame that shows item IDs, the number of score points, expected scores (both from the `paramTab` arguments), as well as the occupied score points

- `formula` the formula passed to `mml`
- `contrasts` the contrasts used in forming the design matrix
- `xlevels` the levels of the covariates used in forming the design matrix
- `polyModel` the value of the argument of the same name passed to `mml`
- `paramTab` a data frame that condenses `dichotParamTab` and `polyParamTab`
- `fast` the value of the argument of the same name passed to `mml`
- `idVar` the value of the argument of the same name passed to `mml`
- `posteriorEsts` the posterior estimates for the people in `stuDat` included in the model
- `pred` the predicted outcome based on the estimated coefficients
- `values` a list of values used for posterior estimation
- `V` a diagonal matrix of the number of columns in the design matrix

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 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 covariance matrix
- `idVar` the name of the identifier used on `stuDat` and `stuItems` data
- `res1` 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 PSU variable on `stuDat`; potentially used for variance estimation

- `stuItems` the data frame passed to `mml` reformatted for use in `mml`
- `formula` the formula passed to `mml`
- `contrasts` the contrasts used in forming the design matrix
- `xlevels` the levels of the covariates used in forming the design matrix
- `polyModel` the value of the argument of the same name passed to `mml`
- `posteriorEsts` the list of posterior estimates for the people in `stuDat` included in the model
- `SubscaleVC` the matrix of latent correlations across constructs
- `modelFrameFull` the full model frame
- `stuDatSubsets` a logical matrix with a column per subscale and a row for each student; used for subsetting students within subscales
- `values` a list of values used for posterior 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("3p1", 11))

# param tab for GPCM items
polyParamTab <- data.frame(ItemID = factor(c("m0757c1", "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(3L, 2L)) # 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 <- EdSurvey::getData(data=sdf,
                        varnames=c(items, "dsex", "origwt", "repgrp1", "jkunit", "sdracem"),
                        dropOmittedLevels = 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("Incorrect", edf[,rawcol]), 0, edf[,coli])
    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 m0757c1
    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
} # end scoreing

# stuDat is one row per student an contains student-level information
stuDat <- edf[,c("sid", "origwt", "repgrp1", "jkunit", "dsex", "sdracem", items)]

# 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,
            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)

# estimate a regression for Numeracy subscale
mmlN <- mml(num ~ dsex,
            stuDat=stuDat,
            dichotParamTab=dichotParamTab, polyParamTab=polyParamTab,
            testScale=testDat,
            Q=128,
            idVar="sid", weightVar="origwt", # these are column names on stuDat
            strataVar="repgrp1", PSUVar="jkunit")
# summary, with Taylor standard errors
summary(mmlN)

# draw plausible values for mmlA
pvd <- drawPVs(summary(mmlA))
head(pvd)

# composite regression
mmlC <- mml(composite ~ dsex ,
            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
mmlCsum <- summary(mmlC)
mmlCsum # show results

# draw plausible values for mmlC, show first few rows
pvc <- drawPVs(mmlCsum)
head(pvc)

## End(Not run)

```

Index

drawPVs, [2](#)

mm1, [3](#), [4](#)