Package: FactorHet (via r-universe)

January 14, 2025

Title Estimate Heterogeneous Effects in Factorial Experiments Using Grouping and Sparsity

Version 1.0.0

Date 2025-01-08

Description Estimates heterogeneous effects in factorial (and conjoint) models. The methodology employs a Bayesian finite mixture of regularized logistic regressions, where moderators can affect each observation's probability of group membership and a sparsity-inducing prior fuses together levels of each factor while respecting ANOVA-style sum-to-zero constraints. Goplerud, Imai, and Pashley (2024) <doi:10.48550/ARXIV.2201.01357> provide further details.

Depends R (>= 3.4.0)

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.2

LinkingTo Rcpp, RcppEigen (>= 0.3.3.4.0)

Imports Rcpp (>= 1.0.1), Matrix, ggplot2, ParamHelpers, mlr, mlrMBO, smoof, lbfgs, methods, utils, stats

Suggests FNN, RSpectra, mclust, ranger, tgp, testthat, covr, tictoc

LazyData true

URL https://github.com/mgoplerud/FactorHet

BugReports https://github.com/mgoplerud/FactorHet/issues

Config/pak/sysreqs libgdal-dev gdal-bin libgeos-dev libglu1-mesa-dev libgmp3-dev libgsl0-dev jags libicu-dev libxml2-dev libmpfr-dev libopenmpi-dev libproj-dev

Repository https://mgoplerud.r-universe.dev

RemoteUrl https://github.com/mgoplerud/factorhet

RemoteRef HEAD

RemoteSha eaa6e84cb05064c7d86c9d73b16b23ae090bb4e9

Contents

AME *Calculate marginal effects*

Description

Calculate the average marginal (component) effect (AME or AMCE), the average combination effect (ACE), or the average marginal interaction effect (AMIE) with a FactorHet model.

Usage

```
AME(
  object,
  baseline = NULL,
  vcov = TRUE,
  design = NULL,
  ignore_restrictions = FALSE,
  vcov.type = NULL,
  average_position = TRUE,
  verbose = TRUE,
  plot = TRUE
)

manual_AME(
  object,
  baseline,
  vcov = TRUE,
  design = NULL,
  extra_restriction = NULL,
  ignore_restrictions = FALSE,
```

```
  vcov.type = NULL,
  average_position = TRUE,
  verbose = TRUE,
  plot = TRUE
)

ACE(
  object,
  baseline,
  design = NULL,
  average_position = TRUE,
  ignore_restrictions = FALSE,
  extra_restriction = NULL,
  verbose = TRUE,
  plot = TRUE
)

AMIE(
  object,
  design = NULL,
  baseline = NULL,
  average_position = TRUE,
  ignore_restrictions = FALSE,
  verbose = FALSE,
  plot = TRUE
)
```

Arguments

object	An object from FactorHet or FactorHet_mbo.
baseline	A named list consisting of each factor and a corresponding baseline level. The default (NULL) computes the effect for all factors using the first level as the baseline. NA uses no baseline and approximates the "marginal means" from Leeper et al. (2020).
vcov	A logical value indicating whether the standard errors for the AME should be computed. The default is TRUE. Standard errors are not implemented for the AMIE.
design	A dataset used to estimate the marginal effects. The default, NULL, uses the estimation data.
ignore_restrictions	A logical value about whether to ignore randomization restrictions when calculating the marginal effects. The default is FALSE. "Details" provides more information.
vcov.type	A string indicating the type of standard errors to be computed. The default is NULL and uses the default settings in vcov.FactorHet; options are specified by that function's se.method argument.

`average_position`
: A logical value indicating whether, for forced choice designs, the "left" and "right" profiles should be averaged. The default is `TRUE`. Goplerud et al. (2025) provide discussion of this point.

`verbose`
: A logical value as to whether more information should be provided on the progress of estimating the effects. The default is `TRUE`.

`plot`
: A logical value as to whether the function should print the plot immediately or quietly provide an object containing the plot and data. The default is `TRUE`.

`extra_restriction`
: A list of additional restrictions to include when computing the marginal effects. The default is `NULL`, i.e. no additional restrictions. "Details" provides more information about the use of this function.

Details

Choice of Baseline: For ACE and AMIE, a choice of baseline is required. See Egami and Imai (2019) for details. For AME, a choice of baseline corresponds to a "standard" AME (see Egami and Imai 2019). The option `NULL` chooses the first level of each factor. It can be manually specified using a named list. If a named list is provided, only AMEs for those named factors are calculated. This can be helpful if there are many factors.

If `NA` is provided as the baseline level, the AME is calculated without a baseline; while this does not correspond to a "proper" AME, it is designed to approximate the "marginal means" discussed in Leeper et al. (2020). Note that in the presence of randomization restrictions, the quantity estimated with a `NA` baseline may not be centered around 0.5. Ignoring the randomization restrictions may be useful in this scenario. Supporting information from Goplerud et al. (2025) provides more discussion of this point.

Randomization Restrictions: Randomization restrictions can be set in one of two ways. By default, FactorHet checks whether for each pairwise combinations of factors, some combination of levels do not occur at all (e.g. "doctor" and "high school") or whether some included interactions are extremely rare (see `rare_threshold` in `FactorHet_control`). Those are assumed to be the randomization restrictions implied by the design. Setting `rare_threshold = 0` forces the inclusion of all interaction terms.

If this procedure for automatically generating randomization restrictions is inappropriate for a specific dataset, randomization restrictions can be set manually as follows using the `manual_AME` function. First, set `ignore_restrictions = TRUE`. This will ignore all "data-driven" estimates of randomization restrictions. Second, the argument `extra_restriction` should be a named list where the name of each element corresponds to a factor (e.g. "Job") and each element is a vector of the levels that *cannot* be used. When using this approach, `AME` should be used only for one factor at a time. An example is shown below.

Plots: Note that for the ggplot2 visualizations of the ACE and AMIE, gray squares indicate combinations that are excluded due to randomization restrictions. White indicates baseline levels.

Value

Returns a named list with the underlying data (`"data"`) and the plot (`"plot"`).

References

Egami, Naoki and Kosuke Imai. 2019. "Causal Interaction in Factorial Experiments: Application to Conjoint Analysis." *Journal of the American Statistical Association*. 114(526):529-540.

Goplerud, Max, Kosuke Imai, and Nicole E. Pashley. 2025. "Estimating Heterogeneous Causal Effects of High-Dimensional Treatments: Application to Conjoint Analysis." arxiv preprint: https://arxiv.org/abs/2201.01357

Leeper, Thomas J., Sara B. Hobolt, and James Tilley. 2020. "Measuring Subgroup Preferences in Conjoint Experiments." *Political Analysis*. 28(2):207-221.

Examples

```
data(immigration)
# Induce "fake" randomization restriction
immigration$joint_id <- paste(immigration$CaseID, immigration$contest_no)
remove_profiles <- subset(immigration, Plans == 'No plans' & Ed == 'GradDeg')
immigration <- subset(immigration, !(joint_id %in% remove_profiles$joint_id))
# Fit with one group and limited regularization for example only
fit <- FactorHet(Chosen_Immigrant ~ Plans + Ed + Country,
  design = immigration, lambda = 1e-4,
  K = 1, group = ~ CaseID, task = ~ contest_no, choice_order = ~ choice_id)
# Estimate AME of "Ed" with randomization restriction
est_AME <- AME(fit, baseline = list('Ed' = 'GradDeg'))

# Estimate AME ignoring randomization restriction
est_AME_norr <- AME(fit,
  baseline = list('Ed' = 'GradDeg'), ignore_restrictions = TRUE)
# Estimate AME by manually specifying randomization restrictions
# this uses the 'manual_AME' function
est_AME_rr_manual <- manual_AME(fit,
  baseline = list('Ed' = 'GradDeg'), ignore_restrictions = TRUE,
  extra_restriction = list('Plans' = 'No plans'))
stopifnot(isTRUE(all.equal(est_AME$data, est_AME_rr_manual$data)))
# Estimate without baseline
est_MM <- AME(fit, baseline = list('Ed' = NA))

# Estimate ACE and AMIE

est_ACE <- ACE(fit, baseline = list('Ed' = 'GradDeg', 'Plans' = 'Has contract'))

est_AMIE <- AMIE(fit, baseline = list('Ed' = 'GradDeg', 'Plans' = 'Has contract'))
```

`cjoint_plot` *Plot a FactorHet object*

Description

Plots the coefficients β_k from a fitted FactorHet object for the main effects only. Use `AME` to calculate average marginal effects. This provides a fast method to examine the impact of a factor.

Usage

```
cjoint_plot(object, baseline = NA, plot = TRUE)
```

Arguments

object	An object from FactorHet or FactorHet_mbo.
baseline	A specification of baseline levels of each factor. The default is NA and shows all coefficients. The documentation for AME discusses how a named list could be provided to show the difference between coefficients.
plot	A logical value as to whether the function should print the plot immediately or quietly provide an object containing the plot and data. The default is TRUE.

Value

Returns a named list with the underlying data ("data") and the plot ("plot").

See Also

AME

Examples

```
# Fit with one group and limited regularization for example only
# Ignore conjoint structure for simplicity
fit <- FactorHet(Chosen_Immigrant ~ Plans + Ed + Country,
  design = immigration, lambda = 1e-2,
  K = 1, group = ~ CaseID, task = ~ contest_no, choice_order = ~ choice_id)
# Plot the raw coefficients
cjoint_plot(fit)
```

---

diff_AME *Difference between AMEs in each group*

Description

Computes the differences between AME between two groups with an accompanying standard error.

Usage

```
diff_AME(object, AME, baseline_group, plot = TRUE)
```

Arguments

object	An object from FactorHet or FactorHet_mbo.
AME	An object containing the average marginal effects estimated using AME.
baseline_group	An integer that denotes the baseline group. The function will show the difference in AMEs from this group, i.e. Group k - Group baseline_group.
plot	A logical value as to whether the function should print the plot immediately or quietly provide an object containing the plot and data. The default is TRUE.

Value

Returns a named list with the underlying data ("data") and the plot ("plot").

Examples

```
# Fit with two groups and fixed lambda for quick illustration

data(immigration)
set.seed(15)
fit <- FactorHet(formula = Chosen_Immigrant ~ Country + Ed,
  design = immigration, group = ~ CaseID, task = ~ contest_no,
  choice_order = ~ choice_id, lambda = 1e-2,
  control = FactorHet_control(init_method = 'random_member'),
  K = 2)
fit_AME <- AME(fit)
diff_AME(fit, fit_AME, baseline_group = 2)
```

FactorHet *Estimate heterogeneous effects in factorial and conjoint experiments*

Description

Fit a model to estimate heterogeneous effects in factorial or conjoint experiments using a "mixture of experts" (i.e. a finite mixture of regularized regressions with covariates affecting group assignment). Effects are regularized using an overlapping group LASSO. FactorHet_mbo finds an optimal lambda via Bayesian optimization whereas FactorHet requires a lambda to be provided. FactorHet_mbo typically used in practice.

Usage

```
FactorHet(
  formula,
  design,
  K,
  lambda,
  moderator = NULL,
  group = NULL,
```

```
  task = NULL,
  choice_order = NULL,
  weights = NULL,
  control = FactorHet_control(),
  initialize = FactorHet_init(),
  verbose = TRUE
)

FactorHet_mbo(
  formula,
  design,
  K,
  moderator = NULL,
  weights = NULL,
  group = NULL,
  task = NULL,
  choice_order = NULL,
  control = FactorHet_control(),
  initialize = FactorHet_init(),
  mbo_control = FactorHet_mbo_control()
)
```

Arguments

formula	Formula specifying model. The syntax is y ~ X1 + X2 where y is the outcome and X1 and X2 are factors. Interactions can be specified using * syntax. All main factors must be explicitly included.
design	A data.frame containing the data to be analyzed.
K	An integer specifying the number of groups; K=1 specifies a model with a single group.
lambda	A positive numeric value denoting regularization strength; this is scaled internally by the number of observations, see FactorHet_control. FactorHet_mbo calibrates through model-based optimization. "Details" provides more discussion of this approach.
moderator	A formula of variables (moderators) that affect the prior probability of group membership. This is ignored when K=1 or moderator=NULL.
group	A formula of a single variable, e.g. ~ person_id, that is used when there are repeated observations per individual.
task	A formula of a single variable that indicates the task number performed by each individual. This is not used when group is unspecified.
choice_order	A formula of a single variable that indicates which profile is on the "left" or "right" in a conjoint experiment.
weights	A formula of a single variable that indicates the weights for each observation (e.g., survey weights). If group is specified, the weights must be constant inside of each value of group.
control	An object from FactorHet_control that sets various model estimation options.

initialize	An object from FactorHet_init that determines how the model is initialized.
verbose	A logical value that prints intermediate information about model fitting. The default is TRUE.
mbo_control	A list of control parameters for MBO; see FactorHet_mbo_control for more information.

Details

Caution: Many settings in FactorHet_control can be modified to allow for slight variations in how the model is estimated. Some of these are faster but may introduce numerical differences across versions of R and machines. The default settings aim to mitigate this. One of the default settings (FactorHet_control(step_SQUAREM=NULL)) considerably increases the speed of convergence and the quality of the optimum located at the expense of sometimes introducing numerical differences across machines. To address this, one could not use SQUAREM (do_SQUAREM=FALSE) or set it to use some fixed step-size (e.g., step_SQUAREM=-10). If SQUAREM produces a large step, a message to this effect will be issued.

Factorial vs. Conjoint Experiment: A factorial experiment, i.e. without a forced-choice between profiles, can be modeled by ignoring the choice_order argument and ensuring that each group and task combination corresponds to exactly one observation in the design.

Estimation: All models are estimated using an AECM algorithm described in Goplerud et al. (2025). Calibration of the amount of regularization (i.e. choosing λ), should be done using FactorHet_mbo. This uses a small number (default 15) of attempts to calibrate the amount of regularization by minimizing a user-specific criterion (defaulting to the BIC), and then fits a final model using the λ that is predicted to minimize the criterion.

Options for the model based optimization (mbo) can be set using FactorHet_mbo_control. Options for model estimation can be set using FactorHet_control.

Ridge Regression: While more experimental, ridge regression can be estimated by setting lambda = 0 (in FactorHet) and then setting prior_var_beta in FactorHet_control or by using FactorHet_mbo and setting mbo_type = "ridge".

Moderators: Moderators can be provided via the moderator argument. These are important when K > 1 for ensuring the stability of the model. Repeated observations per individual can be specified by group and/or task if relevant for a force-choice conjoint.

Value

Returns an object of class FactorHet. Typical use will involve examining the patterns of estimated treatment effects. cjoint_plot shows the raw (logistic) coefficients.

Marginal effects of treatments (e.g. average marginal effects) can be computed using AME, ACE, or AMIE.

The impact of moderators on group membership can be examined using margeff_moderators or posterior_by_moderators.

The returned object is a list containing the following elements:

parameters: Estimated model parameters. These are usually obtained via coef.FactorHet.

K: The number of groups

posterior: Posterior group probability for each observation. This is list of two data.frames one with posterior probabilities (`"posterior"`) and one (`"posterior_predictive"`) implied solely by the moderators, i.e. $\pi_k(X_i)$ from Goplerud et al. (2025).

information_criterion: Information on the BIC, degrees of freedom, log-likelihood, and number of iterations.

internal_parameters: A list of many internal parameters. This is used for debugging or by other post-estimation functions.

vcov: Named list containing the estimated variance-covariance matrix. This is usually extracted with `vcov`.

lp_shortEM: If `"short EM"` is applied (only applicable if `FactorHet`, not `FactorHet_mbo`, is used), it lists the log-posterior at the end of each short run.

MBO: If `FactorHet_mbo` is used, information about the model-based optimization (MBO) is stored here. `visualize_MBO` provides a quick graphical summary of the BIC at different λ.

Examples

```
# Use a small subset of the immigration data from Hainmueller and Hopkins
data(immigration)

set.seed(1)
# Fit with two groups and tune regularization via MBO
fit_MBO <- FactorHet_mbo(
  formula = Chosen_Immigrant ~ Country + Ed + Gender + Plans,
  design = immigration, group = ~ CaseID,
  task =  ~ contest_no, choice_order = ~ choice_id,
  # Only do one guess after initialization for speed
  mbo_control = FactorHet_mbo_control(iters = 1),
  K = 2)
# Plot the raw coefficients
cjoint_plot(fit_MBO)
# Check how MBO fared at calibrating regularization
visualize_MBO(fit_MBO)
# Visualize posterior distribution of group membership
posterior_FactorHet(fit_MBO)
# Get AMEs
AME(fit_MBO)
```

FactorHet-class *Generic methods for FactorHet models*

Description

Brief descriptions of generic methods (e.g. print, summary) for FactorHet as well as a way to visualize the progress of the model-based optimization.

Usage

```
## S3 method for class 'FactorHet'
plot(x, y = NULL, ...)

## S3 method for class 'FactorHet'
formula(x, ...)

## S3 method for class 'FactorHet'
print(x, fusion.tolerance = 0.001, ...)

## S3 method for class 'FactorHet'
summary(object, show_interactions = FALSE, digits = 3, ...)

## S3 method for class 'FactorHet'
coef(object, coef_type = "beta", ...)

## S3 method for class 'FactorHet'
logLik(object, type = "loglik", ...)

## S3 method for class 'FactorHet'
BIC(object, ...)

## S3 method for class 'FactorHet'
AIC(object, ...)

## S3 method for class 'FactorHet_vis'
print(x, ...)

visualize_MBO(object)

posterior_FactorHet(object)

## S3 method for class 'FactorHet'
vcov(object, phi = TRUE, se.method = NULL, ...)
```

Arguments

x	Model from FactorHet
y	Not used; required to maintain compatibility.
...	Optional arguments; only used by `plot.FactorHet` with `cjoint_plot`.
fusion.tolerance	Threshold at which to declare levels fused
object	Object fit using `FactorHet` or `FactorHet_mbo`.
show_interactions	Used by `summary.FactorHet`; indicates whether the interaction terms be shown. Default `FALSE`. See "Details" for more discussion.
digits	Number of digits to include

coef_type	Type of coefficient (beta for treatment effects; phi for moderators)
type	For "logLik", should the log-likelihood ("loglik"), log-posterior ("log_posterior"), or sequence of log-posterior values at each iteration ("log_posterior_seq") be returned?
phi	A logical value indicating whether the standard errors from the moderator parameters, ϕ, should be returned as well. The default is TRUE.
se.method	A string value for the type of standard errors to be computed. The default, and primary option, is NULL which is generally equivalent to "louis" (Louis 1982), as discussed in Goplerud et al. (2025).

Details

The following methods with the arguments given above exist. All methods work on models with using FactorHet and FactorHet_mbo.

plot: This is a shorthand for cjoint_plot on a fitted object.

formula: This returns the underlying formula for the treatment effects and moderators as a named list. This also returns the values used for group, task, and choice_order if provided.

print: This consists of two print methods. For FactorHet, it summarizes the model and fusion of the factor levels. fusion.tolerance sets the threshold at which levels are reported as fused. For outputs of AME (and similar), this plots the corresponding plot. See that documentation for more details.

summary: This summarizes the main effects by group with standard errors. It is typically more common to visualize this with cjoint_plot (and the accompanying data.frame) or AME. show_interactions = TRUE shows the interactions in addition to the main effects.

coef: This returns the coefficient matrix on the original scale (i.e. with the sum-to-zero constraints). code_type = "phi" returns the moderator coefficients instead of the treatment effect coefficients.

AIC and BIC: This returns the AIC or BIC. If multiple degrees of freedom options specified, it returns a matrix.

logLik: This returns the log-likelihood, log-posterior or sequence of log-posterior values at each iteration of the algorithm. The argument "type" provides more details.

visualize_MBO: For a model fit with FactorHet_mbo, this shows information about the MBO, i.e. proposed values and objectives.

posterior_FactorHet: For a model with K > 1, this visualizes the posterior for each observation and the posterior predictive implied by the moderators.

vcov.FactorHet This extracts the estimated variance-covariance matrix of the parameters.

Value

Returns the corresponding output of the generic method. "Details" provides details on the output of each function.

References

Louis, Thomas A. 1982. "Finding the Observed Information Matrix when Using the EM Algorithm." *Journal of the Royal Statistical Society. Series B (Methodological)*. 44(2):226-233.

Goplerud, Max, Kosuke Imai, and Nicole E. Pashley. 2025. "Estimating Heterogeneous Causal Effects of High-Dimensional Treatments: Application to Conjoint Analysis." arxiv preprint: https://arxiv.org/abs/2201.01357

`FactorHet_control` *Control for FactorHet estimation*

Description

Provides a set of control arguments to `FactorHet`. Arguments around the initialization of the model (important when K > 1) can be set via `FactorHet_init` and arguments for the model-based optimization tuning of regularization strength λ can be found in `FactorHet_mbo_control`. The parameters can be divided into ones governing the model priors, model estimation, and miscellaneous settings. All arguments have default values.

Usage

```
FactorHet_control(
  iterations = 1000,
  maxit_pi = NULL,
  optim_phi_controls = list(method = "lib_lbfgs"),
  prior_var_phi = 4,
  prior_var_beta = Inf,
  gamma = 1,
  repeat_beta = 1,
  adaptive_weight = "B&R",
  init_method = "short_EM",
  return_data = FALSE,
  log_method = "log_ginv",
  tolerance.parameters = 1e-05,
  tolerance.logposterior = 1e-05,
  rare_threshold = 5,
  rare_verbose = 1,
  beta_method = "cpp",
  beta_cg_it = 25,
  lambda_scale = "N",
  weight_dlist = FALSE,
  do_SQUAREM = TRUE,
  step_SQUAREM = NULL,
  backtrack_SQUAREM = 10,
  df_method = "EM",
  forced_randomize = FALSE,
  single_intercept = NULL,
```

```
  tau_method = "nullspace",
  tau_stabilization = 5,
  tau_truncate = 1e+06,
  debug = FALSE,
  force_reset = FALSE,
  calc_df = TRUE,
  calc_se = TRUE,
  quiet_tictoc = TRUE,
  override_BR = FALSE
)
```

Arguments

iterations	A numerical value setting the maximum number of iterations used in the algorithm. The default is 1000.
maxit_pi	An argument setting the maximum number of iterations used in each M-Step that updates the moderators. The default is NULL and uses default settings in optimizer. For "lib_lbfgs", this optimizes until convergence is obtained.
optim_phi_controls	A list of options for optimizer used in updating the moderator parameters. A method must be provided at minimum, e.g., list(method = "lib_lbfgs"). "lib_lbfgs" uses lbfgs from the accompanying package. All other options use the base optim function in R. The maximum number of iterations should be specified via maxit_pi. All other options are specified through this argument.
prior_var_phi	A numerical value that encodes the variance of multivariate normal prior on moderator coefficients. **Note:** The moderators are not standardized internally and thus should be on broadly comparable scales to avoid differential amounts of regularization on different moderators. The default value is 4.
prior_var_beta	A numerical value of normal prior on each treatment effect coefficient. The default is Inf when using sparse estimation. A different value can be set when using "ridge" regression, i.e. lambda=0.
gamma	A non-negative numerical value that determines whether sparsity-inducing prior be "spread" across groups in proportion to the average prior probability of membership. Default of 1; see Städler et al. (2010) and Goplerud et al. (2025) for more discussion.
repeat_beta	An integer setting the number of times to repeat the E-M cycle for updating β before moving to update the moderator parameters ϕ. The default is 1.
adaptive_weight	An argument that determines the weights given to different terms in the penalty function. The default ("B&R") uses Bondell and Reich (2009), generalized appropriately if needed, see Goplerud et al. (2025) for discussion. If a matrix is provided (e.g. from a prior run of FactorHet), this can be used to set up an "adaptive overlapping group LASSO". "none" imposes no weights. To use a matrix and *not* use Bondell and Reich weights, additional set override_BR = TRUE.
init_method	An argument for initializing the algorithm. One set of options are different character values: "kmeans" (k-means clustering on the moderators), "mclust"

("mclust" on the moderators), "random_pi" (random probabilities of group membership for each person), "random_member" (random hard assignment), "random_beta" (random coefficients). This can be set with a named list with group membership probabilities. This should consist of a named list with a single element "group_E.prob" that is a data.frame which contains probabilities for each group/unit with the column names "group" and then "group_[0-9]+" depending on K. In general, when using FactorHet_mbo, this argument is not used and rather set via the relevant options in FactorHet_mbo_control as this will ensure the same initialization for all runs of FactorHet_mbo.

return_data A logical value for whether the formatted data should be returned. The default is FALSE.

log_method An argument for specifying whether latent overlapping groups should be used when interactions are included. The default is "log_ginv". Options beginning with "log_" employ latent overlapping groups (see Yan and Bien 2017 and the supporting information of Goplerud et al. 2025). The projection matrix can be either the generalized inverse extending Post and Bondell (2013) ("log_ginv"), a random matrix ("log_random"), or zero ("log_0"). "standard" does not implement overlapping groups.

tolerance.parameters
A numerical value setting the one convergence criterion: When no parameter changes by more than this amount, terminate the algorithm. Default is 1e-5.

tolerance.logposterior
A numerical value setting the one convergence criterion: When the log-posterior changes by less than this amount, terminate the algorithm. Default is 1e-5.

rare_threshold A numerical value setting the threshold for which interactions should be excluded. If an interaction of two factors has fewer than rare_threshold observations, the corresponding interaction term will not be included. This is a way to enforce randomization restrictions. The default is 5 but setting it to 0 will ensure that all interactions are included. The documentation of FactorHet provides more discussion.

rare_verbose A logical value as to whether to print information about the rare interactions. The default is TRUE.

beta_method A character value for the method by which β is updated. The default is "cpp". An alternative that uses conjugate gradient ("cg") is faster per-iteration but may introduce numerical differences across platforms.

beta_cg_it A numerical value of the number of conjugate gradient steps to use if beta_method = "cg".

lambda_scale A function for internally rescaling lambda to be a function of N. Options are "N" (default; lambda * N), "unity" (i.e. no rescaling), or "root_N" (lambda * sqrt(N)).

weight_dlist A logical value for whether to weight additional penalties following Hastie and Lim (2015). The default is FALSE.

do_SQUAREM A logical value for whether to perform SQUAREM to accelerate convergence. The default is TRUE.

`step_SQUAREM` An argument specifying the step size to use for SQUAREM. The default is `NULL` which uses a data-driven step size. This generally performs well, but may introduce numerical differences across machines. See the documentation of `FactorHet` for more discussion.

`backtrack_SQUAREM`
An integer that sets the number of backtracking steps to perform for SQUAREM. The default is 10.

`df_method` A character value specifying the method calculating degrees of freedom. Default of `"EM"` follows Goplerud et al. (2025) and calculates the degrees of freedom using the Polya-Gamma weights. `"IRLS"` uses $\zeta_{ik}(1-\zeta_{ik})$ as weights, where $\zeta_{ik} = Pr(y_i = 1|X_i, z_i = k)$. `"free_param"` counts the number of parameters after fusion and accounting for the sum-to-zero constraints. Use `"all"` to estimate all methods and compare.

`forced_randomize`
A logical value that indicates, in the forced-choice setting, whether the "left" and "right" profiles should be randomized for each task. The default is `FALSE`.

`single_intercept`
A logical value or `NULL` that indicates whether a single intercept should be used across groups. The default is `NULL` which uses a single intercept if the study is a forced-choice conjoint (i.e., `choice_order` is used) and a varying intercept by group otherwise.

`tau_method` A character value indicating the method for dealing with binding restrictions, i.e. numerically infinite $E[1/\tau^2]$. The two options are `"nullspace"` (i.e. perform inference assuming this restriction binds) or `"clip"` (set to a large value `tau_truncate`). The default is `"nullspace"`.

`tau_stabilization`
An integer value of the number of steps to perform with `tau_method="clip"` before using the provided setting. The default is 5.

`tau_truncate` A numerical value to either truncate $E[1/\tau^2]$ (i.e. set maximum $E[1/\tau^2]$ in the E-Step for updating β) if `tau_method = "clip"` or a threshold by which to declare that two levels are fused if `tau_method="nullspace"`. The default is 1e6.

`debug` A logical value for whether the algorithm should be debugged. The default is `FALSE`. In particular, it will verify that the log-posterior increases at each (intermediate) step and throw an exception otherwise.

`force_reset` A logical argument about how the nullspace is computed. If `tau_method="nullspace"`, it forces nullspace to be estimated directly from all binding restrictions at each iteration versus the default method that updates the existing basis when possible. Default is `FALSE`.

`calc_df` A logical value for whether to calculate degrees of freedom of final model. The default is `TRUE`.

`calc_se` A logical value for whether standard errors of final model. The default is `TRUE`.

`quiet_tictoc` A logical value for whether to *not* print information about the timing of the model. The default is `TRUE`.

`override_BR` A logical value for whether to ignore Bondell and Reich style-weights. The default is `FALSE`. If `TRUE` is provided, `sqrt(L) * (L + 1)` is used, where `L` is the number of factor levels.

Value

`FactorHet_control` returns a named list containing the elements listed in "Arguments".

References

Bondell, Howard D., and Brian J. Reich. 2009. "Simultaneous Factor Selection and Collapsing Levels in ANOVA." Biometrics 65(1): 169-177.

Goplerud, Max, Kosuke Imai, and Nicole E. Pashley. 2025. "Estimating Heterogeneous Causal Effects of High-Dimensional Treatments: Application to Conjoint Analysis." arxiv preprint: https://arxiv.org/abs/2201.01357

Post, Justin B., and Howard D. Bondell. 2013. "Factor Selection and Structural Identification in the Interaction ANOVA Model." *Biometrics* 69(1):70-79.

Lim, Michael, and Trevor Hastie. 2015. "Learning Interactions via Hierarchical Group-Lasso Regularization." *Journal of Computational and Graphical Statistics* 24(3):627-654.

Städler, Nicolas, Peter Bühlmann, and Sara Van De Geer. 2010. "l1-penalization for Mixture Regression Models." *Test* 19(2):209-256.

Yan, Xiaohan and Jacob Bien. 2017. "Hierarchical Sparse Modeling: A Choice of Two Group Lasso Formulations." *Statistical Science* 32(4):531–560.

Examples

```
str(FactorHet_control())
```

`FactorHet_init` *Arguments for initializing FactorHet*

Description

A set of arguments that govern the initialization of `FactorHet`. Use `FactorHet_control` to set arguments around estimation. `FactorHet_mbo` ignores many of these arguments as it uses a single fixed initialization set by `FactorHet_mbo_control`. All arguments have default values.

Usage

```
FactorHet_init(
  short_EM = FALSE,
  short_EM_it = 40,
  short_EM_init = "random_member",
  short_EM_pi = NULL,
  force_rep = FALSE,
  verbose = FALSE,
  short_EM_beta_method = "cpp",
  short_EM_cg_it = 10,
  nrep = 5,
```

```
  debug_repeat = FALSE,
  plot_repeat = FALSE,
  return_all = FALSE
)
```

Arguments

short_EM	A logical value indicating whether "short EM" should be used. The default value is FALSE. TRUE indicates a "short EM" should be followed. That is, run multiple short runs of EM with random initializations and then proceed with the best for full initialization. Biernacki et al. (2003) provides more discussion. If FactorHet_control has init_method = "short_EM", this will override this setting.
short_EM_it	A numerical value of the number of iterations to use for each "short" run of the EM algorithm. The default is 40.
short_EM_init	An argument that sets the initialization procedure for "short EM". It must be some non-deterministic procedure that is valid in FactorHet_control. The default is "random_member".
short_EM_pi	An argument for the maximum number of iterations for the moderator updates to use for each "short" run of the EM algorithm. The default is NULL.
force_rep	A logical value for whether to repeat the algorithm if K=1. The default is FALSE and it should be used only for debugging.
verbose	A logical value to print more information about the progress of each iteration. The default is FALSE.
short_EM_beta_method	An argument for the update method for β to use for each "short" run of the EM algorithm. The default is "cpp".
short_EM_cg_it	An argument for the number of conjugate gradient iterations to use if short_EM_beta_method = "cg".
nrep	An integer value of the number of random iterations or runs of "short EM" should be used. The default value is 5.
debug_repeat	A logical value for whether to debug the repeated runs. The default is FALSE.
plot_repeat	A logical value for whether to plot the trajectory of the log-posterior for each run. The default is FALSE.
return_all	A logical value for whether to return all repetitions of the model versus the one with the highest log-posterior. The default is FALSE.

Value

FactorHet_init returns a named list containing the elements listed in "Arguments".

References

Biernacki, Christophe, Gilles Celeux, and Gérard Govaert. "Choosing Starting Values for the EM algorithm for Getting the Highest Likelihood in Multivariate Gaussian Mixture Models." 2003. *Computational Statistics & Data Analysis*. 41(3-4):561-575.

Examples

```
str(FactorHet_init())
```

FactorHet_mbo_control *Control for model-based optimization*

Description

FactorHet_mbo_control is used to adjust the settings for the MBO (model-based optimization). All arguments have default values. This relies heavily on options from the mlrMBO package so please see this package for more detailed discussion.

Usage

```
FactorHet_mbo_control(
  mbo_type = c("sparse", "ridge"),
  mbo_initialize = "mm_mclust_prob",
  mm_init_iterations = NULL,
  mbo_range = c(-5, 0),
  mbo_method = "regr.bgp",
  final_method = "best.predicted",
  iters = 11,
  mbo_noisy = TRUE,
  criterion = c("BIC", "AIC", "GCV", "BIC_group"),
  ic_method = c("EM", "IRLS", "free_param"),
  se_final = TRUE,
  mbo_design = -1.5,
  fast_estimation = NULL,
  verbose = FALSE
)
```

Arguments

mbo_type	A character argument indicating the type of model to estimate. The default is "sparse" which uses the structured sparse penalty discussed in Goplerud et al. (2025) and discussed in FactorHet. "ridge" performs a ridge regression.
mbo_initialize	An argument for the initialization method for each MBO proposal. The default is "mm_mclust_prob". "Details" provides a more in-depth discussion.
mm_init_iterations	An integer value of the number of iterations to use if Murphy/Murphy initialization is used. The default is NULL which uses default values of 100 if probabilistic and 50 if deterministic. "Details" provides a more in-depth discussion.
mbo_range	A vector of numerical values that set the range of values to consider on log10(lambda), before standardization (e.g., scaling by N, see FactorHet_control. The default is c(-5,0). "Details" provides more information.

`mbo_method` A function used to propose new values of the regularization parameters. See information from `mlr` for more details. The default is `"regr.bgp"` which requires the `tgp` package to be installed.

`final_method` A character argument that determines how the final regularization parameter should be selected. The default is `"best_predicted"` that uses the regularization parameter that is predicted to have the best value of the criterion. Other options are described in detail in `makeMBOControl` for `final.method`. Alternative options include `"last.proposed"` and `"best.true.y"`.

`iters` A non-negative integer value of the number of proposals to do after initialization. The default is 11.

`mbo_noisy` A logical value for whether to treat the objective function as "noisy" for purposes of model-based optimization. The default is `TRUE`. The `"noisy_optimization"` vignette from `mlrMBO` provides more details. The criterion function is not, in fact, noisy but this option often performs better for a non-smooth function. It uses `link[mlrMBO]{crit.eqi}` instead of `link[mlrMBO]{crit.ei}`.

`criterion` A character value of the criterion to minimize. Options are `"BIC"` (default), `"AIC"`, `"GCV"`, or `"BIC_group"`. `"BIC_group"` counts the number of observations as the number of individuals (e.g., in the case of repeated observations per person).

`ic_method` A character value for the method for calculating degrees of freedom: `"EM"` (default), `"IRLS"`, and `"free_param"`. See `FactorHet_control` for more information.

`se_final` A logical value for whether standard errors be calculated for the final model. The default value is `TRUE`.

`mbo_design` An argument for how to design the initial proposals for MBO. The default is -1.5; this and other options are described in "Details".

`fast_estimation` An argument as to whether a weaker convergence criterion should be used for MBO. The default is `NULL` which uses the *same* arguments for all models. "Details" provides more information.

`verbose` A logical argument to provide more information on the initial steps for MBO; the default is `FALSE`.

Details

Initialization: `FactorHet_mbo` relies on the same initialization for each attempt. The default procedure (`"mm_mclust_prob"`) is discussed in detail in the appendix of Goplerud et al. (2025) and builds on Murphy and Murphy (2020). In brief, it deterministically initializes group memberships using only the moderators (e.g. using `"mclust"`). Using those memberships, it uses an EM algorithm (with probabilistic assignment, if `"prob"` is specified, or hard assignment otherwise) for a few steps with only the main effects to update the proposed group memberships. If the warning appears that "Murphy/Murphy initialization did not fully converge" , this mean that this initial step did not fully converge. The number of iterations could be increased using `mm_init_iterations` if desired, although benefits are usually modest beyond the default settings. These memberships are then used to initialize the model at each proposed regularization value.

The options available are "spectral" and "mclust" that use "spectral" or "mclust" on the moderators with no Murphy/Murphy style tuning. Alternatively, "mm_mclust" and "mm_spectral" combine the Murphy/Murphy tuning upon the corresponding initial deterministic initialization (e.g. spectral or "mclust"). These use hard assignment at each step and likely will converge more quickly although a hard initial assignment may not be desirable. Adding the suffix "_prob" to the "mm_*" options uses a standard (soft-assignment) EM algorithm during the Murphy/Murphy tuning.

If one wishes to use a custom initialization for MBO, then set mbo_initialize=NULL and provide an initialization via FactorHet_control. It is strongly advised to use a deterministic initialization if done manually, e.g. by providing a list of initial assignment probabilities for each group.

Design of MBO Proposals: The MBO procedure works as follows; there are some initial proposals that are evaluated in terms of the criterion. Given those initial proposals, there are iters attempts to improve the criterion through methods described in detail in mlrMBO (Bischl et al. 2018). A default of 11 seems to work well, though one can examine visualize_MBO after estimation to see how the criterion varied across the proposals.

By default, the regularization parameter is assumed to run from -5 to 0 on the log10 scale, before standardizing by the size of the dataset. We found this to be reasonable, but it can be adjusted using mbo_range.

It is possible to calibrate the initial proposals to help the algorithm find a minimum of the criterion more quickly. This is controlled by mbo_design which accepts the following options. Note that a manual grid search can be provided using the data.frame option below.

Scalar: By default, this is initialized with a scalar (-1.5) that is the log10 of lambda, before standardization as discussed in FactorHet_control. For a scalar value, four proposals are generated that start with the scalar value and adjust it based on the level of sparsity of the initial estimated model. This attempts to avoid initializations that are too dense and thus are very slow to estimate, as well as ones that are too sparse.

"random": If the string "random" is provided, this follows the default settings in mlrMBO and generates random proposals.

data.frame: A custom grid can be provided using a data.frame that has two columns ("l" and "y"). "l" provides the proposed values on the log10 lambda scale (before standardization). If the corresponding BIC value is known, e.g. from a prior run of the algorithm, the column "y" should contain this value. If it is unknown, leave the value as NA and the value will be estimated. Thus, if a manual grid search is desired, this can be done as follows. Create a data.frame with the grid values "l" and all "y" as NA. Then, set iters = 0 to do no estimation *after* the grid search.

Estimation: Typically, estimation proceeds using the same settings for each MBO proposal and the final model estimated given the best regularization value (see option final_method for details). However, if one wishes to use a lower convergence criterion for the MBO proposals to speed estimation, this can be done using the fast_estimation option. This proceeds by giving a named list with two members "final" and "fast". Each of these should be a list with two elements "tolerance.logposterior" and "tolerance.parameters" with the corresponding convergence thresholds. "final" is used for the final model and "fast" is used for evaluating all of the MBO proposals.

Value

FactorHet_mbo_control returns a named list containing the elements listed in "Arguments".

References

Bischl, Bernd, Jakob Richter, Jakob Bossek, Daniel Horn, Janek Thomas and Michel Lang. 2018. "mlrMBO: A Modular Framework for Model-Based Optimization of Expensive Black-Box Functions." arxiv preprint: https://arxiv.org/abs/1703.03373

Goplerud, Max, Kosuke Imai, and Nicole E. Pashley. 2025. "Estimating Heterogeneous Causal Effects of High-Dimensional Treatments: Application to Conjoint Analysis." arxiv preprint: https://arxiv.org/abs/2201.01357

Murphy, Keefe and Thomas Brendan Murphy. 2020. "Gaussian Parsimonious Clustering Models with Covariates and a Noise Component." *Advances in Data Analysis and Classification* 14:293–325.

Examples

```
str(FactorHet_mbo_control())
```

`FactorHet_refit` *Refit model using estimated sparsity patterns*

Description

Using a previously estimated model, this function takes the estimated sparsity patterns (e.g., which levels are fused together) and the estimates of the moderator parameters, $\hat{\phi}$, and re-estimates the regression parameters β.

Usage

```
FactorHet_refit(
  object,
  newdata,
  tolerance = 0.001,
  hard_assign = FALSE,
  iter_refit = 200
)
```

Arguments

`object`	An object from `FactorHet` or `FactorHet_mbo`.
`newdata`	A data.frame containing the data to be estimated in the refit model.
`tolerance`	A numerical value that sets the threshold at which to declare two levels as "fused"; the default is 1e-3. Two levels meet this threshold if the maximum difference between the main effects and any interactions is `tolerance`.
`hard_assign`	A logical value that sets whether observations should be be assigned to the most probable cluster given ϕ from the original model or whether they should they be weighted according to their estimated group membership probabilities, $\hat{\pi}(X_i)$. The default is `FALSE` which uses the weighted method.

`iter_refit` An integer value that sets the number of iterations used in fitting the refit model. The default is 200. A warning will be produced if it does not converge in this many iterations.

Details

The main use of this function is to enable sample-splitting as discussed in Goplerud et al. (2025) to improve coverage and remove bias from the initial estimates. An example is provided below.

Value

An object of class `FactorHet` that contains the output described the linked documentation.

Examples

```
data(immigration)
set.seed(1)
# Split the data into two parts for sample-splitting
train_data <- subset(immigration, CaseID < 900)
refit_data <- subset(immigration, CaseID >= 900)
# Fit using fixed lambda for demonstration
# only
fit <- FactorHet(Chosen_Immigrant ~ Plans + Ed + Country,
  design = train_data, lambda = 1e-2,
  moderator = ~ party_ID + census_div,
  control = FactorHet_control(init = 'mclust'),
  K = 2, group = ~ CaseID, task = ~ contest_no, choice_order = ~ choice_id)
# Refit using the other half of data

refit <- FactorHet_refit(fit, newdat = refit_data)
# AME (etc.) for treatment effects can be computed as normal
AME_refit <- AME(refit)
# As can heterogeneous effects, although uncertainty in
# phi is ignored
HTE_refit <- HTE_by_individual(refit, AME_refit, design = immigration)
```

HTE *Estimate heterogeneous treatment effects by individual or moderator*

Description

These functions estimate heterogeneous effects from `FactorHet` at the individual level or by the average value of a moderator. They can be used to produce individual-level estimates that can be compared against other methods for estimating heterogeneous effects.

Usage

```
HTE_by_individual(object, AME, design = NULL)

HTE_by_moderator(
  object,
  AME,
  moderators = NULL,
  design = NULL,
  points_continuous = 10,
  overall_AME = FALSE,
  verbose = FALSE
)
```

Arguments

object	An object from FactorHet, FactorHet_mbo.
AME	An estimate of the average marginal effects by group from AME.
design	An optional data.frame of moderator values on which to produce the individual-level or average conditional average marginal effects. **Note:** There should be one row per observation if this function is used. The default is NULL which uses the estimation data.
moderators	An argument that contains a list of moderators to evaluate. The default is NULL and considers all moderators.
points_continuous	A positive integer value that indicates the number of equally spaced points to evaluate a continuous moderator over.
overall_AME	A logical value that indicates whether to compute the AME over the entire design without modification. The default is FALSE.
verbose	A logical value that indicates whether progress should be reported. The default is FALSE.

Details

The functions here allow for, first, estimation of conditional average marginal effects for each individual given their pre-treatment moderators (HTE_by_individual). This is a weighted average of the AME for each group by the individual's group membership probabilities, i.e. $\hat{\pi}(X_i)$ (Goplerud et al. 2025). These are also averaged together to return an estimate to produce a population-level effect.

Second, one can estimate conditional average marginal effects using HTE_by_moderator. This takes a moderator such as party identification and counterfactually sets each observation to some level (e.g., "Strong Democrat"). It then reports the average of the individual-level conditional effects across the sample population as the "conditional" average marginal effect. If overall_AME is true, it also returns the average of the individual heterogeneous effects given the observed distribution of pre-treatment moderators. It and the population element of the list produced by HTE_by_individual coincide exactly.

Both functions can be used with split-sample or refit, i.e. FactorHet_refit, and the computed AME, although this will not take into account uncertainty in the moderator estimation as they are assumed fixed when refitting the model.

To use these functions, first estimate the AMEs by group, i.e., using AME and then pass this object and the original FactorHet model to the functions for computing heterogeneous effects by moderator or individual.

Value

HTE_by_individual returns a list with two data.frames. The first individual contains the estimated individual conditional average marginal effects. The second population contains the average of those individual effects. Standard errors (via the column var) are also included.

HTE_by_population returns a list for each moderator that consists itself of a list of each value of the moderator used. The value "out" contains the conditional average marginal effects.

Examples

```
data(immigration)
# Estimate model with arbitrary choice of lambda
fit <- FactorHet(Chosen_Immigrant ~ Plans + Ed + Country,
  design = immigration, lambda = 1e-2,
  moderator = ~ party_ID,
  K = 2, group = ~ CaseID,
  control = FactorHet_control(init = 'mclust'),
  task = ~ contest_no, choice_order = ~ choice_id)
# Estimate AME
est_AME <- AME(fit)
# Get individual AME; note only seven distinct
# patterns will exist as partyID is the only (discrete)
# moderator
iAME <- HTE_by_individual(
  object = fit,
  AME = est_AME)
# Get conditional AME by level of party ID
cAME_pID <- HTE_by_moderator(
  object = fit,
  AME = est_AME, overall_AME = TRUE)

AME_1 <- cAME_pID$`Overall AME`$out[[1]][,c('factor', 'level', 'est', 'var')]
AME_2 <- iAME$population[,c('factor', 'level', 'est', 'var')]
rownames(AME_1) <- rownames(AME_2) <- NULL
stopifnot(isTRUE(all.equal(AME_1, AME_2)))
```

---

immigration *Small dataset on immigration preferences*

Description

An example dataset of 100 randomly chosen respondents from the replication data in Hainmueller and Hopkins (2015). Only a small selection of the factors and moderators in the original experiment are included in this example dataset. The full data can be downloaded from the replication archive in "Source" below. The original paper provides more details on all variables. The replication data for Goplerud et al. (2025) provides code to process and analyze the original data using `FactorHet`.

Usage

```
immigration
```

Format

A data frame with 1000 rows and 10 variables:

CaseID Unique identifier for respondent.

contest_no Task number (1-5) for each respondent.

choice_id Identifier for the profile shown, i.e., was it the "left" or "right" profile.

Chosen_Immigrant Immigrant profile chosen by respondent.

Country Immigrant's country of origin.

Ed Immigrant's education level.

Plans Immigrant's employment plans after arrival.

Gender Immigrant's gender.

party_ID **Respondent's** party identification.

census_div Level of immigration in **respondent's** ZIP code.

Source

doi:10.7910/DVN/25505

References

Goplerud, Max, Kosuke Imai, and Nicole E. Pashley. 2025. "Estimating Heterogeneous Causal Effects of High-Dimensional Treatments: Application to Conjoint Analysis." arxiv preprint: https://arxiv.org/abs/2201.01357

Hainmueller, Jens and Daniel J. Hopkins. 2015. "The Hidden American Immigration Consensus: A Conjoint Analysis of Attitudes Toward Immigrants." *American Journal of Political Science* 59(3):529-548.

`margeff_moderators` *Compute association between moderators and group membership*

Description

This function computes the impact of changing a moderator on the group membership probabilities.

Usage

```
margeff_moderators(
  object,
  newdata = NULL,
  vcov = TRUE,
  se.method = NULL,
  quant_continuous = c(0.25, 0.75),
  abs_diff = FALSE
)
```

Arguments

`object`	An object from `FactorHet` or `FactorHet_mbo`.
`newdata`	An optional argument that provides the data over which to average the distribution of the other moderators. The default is `NULL` which uses the estimation data.
`vcov`	A logical value indicating whether the standard errors should be computed. The default is `TRUE`.
`se.method`	An optional argument as to the type of standard errors used. The default is `NULL` uses estimated standard errors. `vcov.FactorHet` provides more information.
`quant_continuous`	A numeric vector consisting of two values between 0 and 1. For continuous moderators, it sets two quantiles of the moderator's distribution to show the difference between. The default `c(0.25,0.75)` compares the effect of changing the moderator from its 25th percentile to its 75th percentile.
`abs_diff`	A logical value as to whether the difference or absolute difference in the change in $\pi_k(X_i)$ should be shown. The default is `FALSE` which returns the standard "marginal effect" of changing the moderators with a standard error computed via the delta method. The value `TRUE` draws 10,000 samples from the asymptotic distribution of the moderators and computes the average the **absolute values** of the marginal effects for each observation in `newdata` using those samples. This is considerably slower than the default setting. The appendix of Goplerud et al. (2025) illustrates one use of this argument.

Details

This function computes the change in $\pi_k(X_i)$ for the change in one of the moderators in X_i. The change is averaged across the distribution of the other moderators found in `newdata` (or, by default, the estimation data). It thus can be thought of as the "marginal effect" of changing one moderator on the probability of group memberships, holding all other moderators constant. It returns a data.frame of the estimated effects as well as a plot to visualize the changes in $\pi_k(X_i)$. Goplerud et al. (2025) provides more discussion of this method.

Value

Returns a named list with the underlying data (`"data"`) and the plot (`"plot"`).

Examples

```
# Estimate model with arbitrary choice of lambda
data(immigration)
set.seed(15)
# Estimate model with arbitrary choice of lambda
fit <- FactorHet(Chosen_Immigrant ~ Plans + Ed + Country,
  design = immigration, lambda = 1e-2,
  moderator = ~ party_ID,
  K = 2, group = ~ CaseID,
  control = FactorHet_control(init = 'mclust'),
  task = ~ contest_no, choice_order = ~ choice_id)
margeff_moderators(fit)
```

---

posterior_by_moderators

Visualize the posterior by observed moderators

Description

Provides univariate summaries of the estimated posterior predictive probabilities of group membership by the moderators. Can produce analyses for continuous variables (weighted boxplot) or discrete variables (row/column tables).

Usage

```
posterior_by_moderators(
  object,
  visualize = c("all", "discrete", "continuous"),
  type_discrete = c("bar", "row", "column", "all")
)
```

Arguments

`object`	A model fit using `FactorHet` or `FactorHet_mbo`.
`visualize`	Specifies which types of moderators to show. Default (`"all"`) shows all moderators. Other options include `"discrete"` and `"continuous"`.
`type_discrete`	Show the results by `"row"` or `"column"` or `"all"` (i.e. both).

Details

Discrete Moderators: Discrete moderators are shown by either a `"row"`, `"column"`, or `"bar"` plot. In the `"row"` plot, the quantity reported is, for each level of the moderator, what proportion of people fall into each group. For example, for moderator value "a", 25% of people are in group 1 and 75% of people are in group 2. This is estimated using a weighted average, weighting by the estimated posterior predictive probabilities of group membership and any survey weights.

By contrast `"column"` and `"bar"` reports the distribution by group. For example, for Group 1, 30% of people have moderator value "f", 50% have moderator value "g", and 20% have moderator value "h". `"bar"` reports this as a bar chart whereas `"column"` reports as a tile plot.

For all three types of plots, the data is provided in the returned output.

Continuous Moderators: Continuous moderators are shown by a histogram of the value for each group, weighted by each observation's posterior predictive probability of being in that group.

Value

A list of each of the types of analyses is reported. Each element of the list contains the ggplot object and the data ("plot" and "data").

Examples

```
data(immigration)
set.seed(15)
# Estimate model with arbitrary choice of lambda
fit <- FactorHet(Chosen_Immigrant ~ Plans + Ed + Country,
  design = immigration, lambda = 1e-2,
  moderator = ~ party_ID,
  K = 2, group = ~ CaseID,
  control = FactorHet_control(init = 'mclust'),
  task = ~ contest_no, choice_order = ~ choice_id)
posterior_by_moderators(fit)
```

---

predict.FactorHet *Predict after using FactorHet*

Description

Predicted values for choosing particular profiles based on `FactorHet`.

Usage

```
## S3 method for class 'FactorHet'
predict(
  object,
  newdata = NULL,
  type = "posterior",
  by_group = FALSE,
  return = "prediction",
  ...
)
```

Arguments

object	A model estimated using FactorHet or FactorHet_mbo.
newdata	A dataset on which to generate predictions. The default, NULL, uses the estimation data.
type	An argument that specifies how to deal with group-membership probabilities when making predictions. The default is "posterior" which use the posterior probabilities for each observation in the training data for weighting the groups. If "posterior_predictive" is used, the group membership probabilities implied by the moderators, i.e. $\hat{\pi}_k(X_i)$, will be used. If an observation in newdata is not in the estimation data (i.e., its value of group) is not found, then "posterior_predictive", i.e. $\hat{\pi}_k(X_i)$, is used.
by_group	A logical value as to whether the predictions should be returned for each group or whether a weighted averaged based on the group membership probabilities (specified by type) should be reported. The default is FALSE.
return	A character value that determines the type of prediction return. The default is "prediction" that returns the predicted probability. The option "detailed" returns a variety of additional information. This is mostly called internally for other functions such as AME or margeff_moderators.
...	Miscellaneous options used internally and not documented.

Value

Returns an estimate of the predicted probability of choosing a profile for each observation. "Arguments" outlines different behavior if certain options are chosen.

Examples

```
data(immigration)
set.seed(1)
# Fit a model once for simplicity
fit <- FactorHet(Chosen_Immigrant ~ Plans + Ed + Country,
 design = immigration, lambda = 1e-4,
 # Randomly initialize, do only one iteration for speed
 init = FactorHet_init(nrep = 1),
 control = FactorHet_control(init = 'random_member'),
 K = 2, group = ~ CaseID, task = ~ contest_no,
```

```
 choice_order = ~ choice_id)
immigration$pred_FH <- predict(fit)
```

Index