Package: idiographic (via r-universe)

July 1, 2026

Type Package

Title Network Estimation from Intensive Longitudinal Data

Version 0.1.0

Author Mohammed Saqr [aut, cre]

Maintainer Mohammed Saqr <mohammed.saqr@uef.fi>

Description Person-specific and within-person network estimation from intensive longitudinal and panel data. Provides preprocessing audits, edge-stability diagnostics, model-comparison reports, rolling forecast validation, rolling ordinary and graphical vector autoregression, ordinary vector autoregression (VAR), graphical vector autoregression (graphical VAR), multilevel vector autoregression (mlVAR), native Bayesian VAR and multilevel VAR that statistically reproduce 'Mplus' Dynamic Structural Equation Modeling (DSEM) output without requiring 'Mplus', unified Structural Equation Modeling (uSEM), and Group Iterative Multiple Model Estimation (GIMME) as clean-room implementations. Split out of the 'Nestimate' package so the idiographic time-series methods carry their own dependencies. Results have tidy accessors and 'cograph_network' plotting support.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.1)

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports stats, utils, lme4, lavaan

Suggests testthat (>= 3.0.0), cograph, mlVAR, MplusAutomation, gimme, graphicalVAR, glasso, qgraph, rio, knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

Config/pak/sysreqs cmake make

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

Date/Publication 2026-07-01 20:58:22 UTC

RemoteUrl https://github.com/mohsaqr/idiographic

RemoteRef HEAD

RemoteSha 8bcb5b760c6830852cb37312e1a2c320986ac0ca

Contents

`as_netobject` *Coerce to a netobject*

Description

Returns netobjects unchanged; promotes a bare `cograph_network`.

Usage

```
as_netobject(x, ...)
```

Arguments

`x`	A `netobject` or `cograph_network`.
`...`	Passed to methods.

Value

A `c("netobject", "cograph_network")` object.

`as_netobject.gvar_result`

Coerce a gvar_result to plottable netobjects

Description

Returns the two networks a graphical VAR contains as Nestimate netobjects, so each renders directly with `cograph::splot()` (or any netobject verb) without the caller transposing matrices or dropping intercept columns. The temporal network is oriented `[from = predictor(t-1), to = outcome(t)]`.

Usage

```
## S3 method for class 'gvar_result'
as_netobject(x, ...)
```

Arguments

`x`	A `gvar_result`.
`...`	Ignored.

Value

A `netobject_group`: a named list with `$temporal` (directed) and `$contemporaneous` (undirected) netobjects.

`as_netobject.net_gimme`

Plottable netobject(s) from a GIMME fit

Description

Returns the GIMME result as matrix-backed netobjects. By default these encode the **same quantity the `gimme` package plots** — the *proportion of subjects* that have each path (`path_counts / n_subjects`) — not the group-average coefficient (which dilutes toward zero and is not what GIMME displays). For the faithful single mixed network (dashed lag / solid contemporaneous, group/individual colouring, autoregressive self-loops) use `plot_gimme()`.

Usage

```
## S3 method for class 'net_gimme'
as_netobject(x, style = c("pnode", "unified"), weight = c("prop", "coef"), ...)
```

Arguments

x	A net_gimme object.
style	Either "pnode" (default) — a netobject_group of two directed p-node networks, $temporal (lagged; autoregression on the diagonal) and $contemporaneous (same-beep), matching the shape graphical_var() returns — or "unified", a single directed 2p-node network with the *_lag half feeding the current half (the literal uSEM topology).
weight	Either "prop" (default) — edge weight is the proportion of subjects with the path — or "coef", the group-average standardized coefficient.
...	Unused.

Value

For style = "pnode", a netobject_group with $temporal and $contemporaneous. For style = "unified", one c("netobject", "cograph_network") object with 2p nodes.

See Also

plot_gimme() for the faithful gimme-style mixed plot.

---

as_netobject.net_mlvar

Plottable netobjects from an mlVAR fit

Description

Returns the three networks as netobjects oriented for plotting (temporal edges run predictor -> outcome, matching graphical_var), so cograph::splot() renders them consistently. The raw fit$temporal$weights keep mlVAR's [outcome, predictor] layout for equivalence.

Usage

```
## S3 method for class 'net_mlvar'
as_netobject(x, ...)
```

Arguments

x	A net_mlvar object.
...	Unused.

Value

A netobject_group with $temporal, $contemporaneous, $between.

as_netobject.var_bayes_result *Coerce a var_bayes_result to plottable netobjects*

Description

Coerce a var_bayes_result to plottable netobjects

Usage

```
## S3 method for class 'var_bayes_result'
as_netobject(x, ...)
```

Arguments

x	A var_bayes_result.
...	Ignored.

Value

A netobject_group with temporal (directed) and contemporaneous (undirected) netobjects.

audit_preprocess *Audit preprocessing and lag construction*

Description

Builds the same lag-1 design used by graphical_var() and build_var(), then returns tidy diagnostics for missingness, day-boundary drops, simple linear trends, AR(1) persistence, split-half mean/variance drift, an ADF-style unit-root screen, and zero-variance variables. This is a preflight tool: it does not fit a network, but it makes the modelling input explicit before estimating VAR, graphical VAR, uSEM, GIMME, or mlVAR models.

Usage

```
audit_preprocess(
  data,
  vars,
  id = NULL,
  day = NULL,
  beep = NULL,
  scale = TRUE,
  center_within = TRUE,
  delete_missings = TRUE,
  min_obs = NULL,
```

```
  subject = NULL,
  trend_alpha = 0.05,
  ar_threshold = 0.95,
  mean_shift_threshold = 0.8,
  sd_ratio_threshold = 2,
  unit_root_t_cutoff = -2.86
)
```

Arguments

data	A data.frame or matrix with columns for variables and optional id/day/beep columns.
vars	Character vector of variable names.
id	Character. Name of the person-ID column, or NULL.
day	Character. Name of the day/session column, or NULL.
beep	Character. Name of the measurement-occasion column, or NULL.
scale	Logical. Whether to standardize variables before lagging.
center_within	Logical. Whether to center within person when more than one id is present.
delete_missings	Logical. If TRUE, $pairs contains only complete current/lagged rows; if FALSE, incomplete rows are retained with NA lags.
min_obs	Integer or NULL. Keep only subjects with at least this many observations.
subject	Optional vector naming the subject(s) to audit.
trend_alpha	Numeric p-value cutoff for the trend flag.
ar_threshold	Numeric absolute AR(1) cutoff for the high-persistence flag.
mean_shift_threshold	Numeric absolute standardized split-half mean shift cutoff.
sd_ratio_threshold	Numeric split-half SD ratio cutoff.
unit_root_t_cutoff	Numeric cutoff for the ADF-style lag-level t-statistic. Values greater than this cutoff are flagged as unit-root risk.

Value

A preprocess_audit object with pairs, counts, diagnostics, and matrices.

See Also

build_var(), graphical_var()

Examples

```
set.seed(1)
d <- data.frame(id = 1, day = 1, beep = 1:40,
                A = rnorm(40), B = rnorm(40))
audit <- audit_preprocess(d, vars = c("A", "B"), id = "id",
                          day = "day", beep = "beep")
audit$counts
audit$diagnostics
```

`build_gimme` *GIMME: Group Iterative Multiple Model Estimation*

Description

Estimates person-specific directed networks from intensive longitudinal data using the unified Structural Equation Modeling (uSEM) framework. Implements a data-driven search that identifies:

1. **Group-level paths**: Directed edges present for a majority (default 75\
2. **Individual-level paths**: Additional edges specific to each person, found after group paths are established.

Uses `lavaan` for SEM estimation and modification indices. Accepts a single data frame with an ID column (not CSV directories).

Usage

```
build_gimme(
  data,
  vars,
  id,
  time = NULL,
  day = NULL,
  beep = NULL,
  min_obs = NULL,
  subject = NULL,
  ar = TRUE,
  standardize = FALSE,
  groupcutoff = 0.75,
  subcutoff = 0.75,
  paths = NULL,
  exogenous = NULL,
  hybrid = FALSE,
  VAR = FALSE,
  rmsea_cutoff = 0.05,
  srmr_cutoff = 0.05,
  nnfi_cutoff = 0.95,
  cfi_cutoff = 0.95,
```

```
  n_excellent = 2L,
  seed = NULL,
  group_correct = "Bonferoni Group",
  subgroup = FALSE,
  outcome = NULL,
  conv_vars = NULL,
  mult_vars = NULL,
  lv_model = NULL,
  lasso_model_crit = NULL,
  ms_allow = FALSE,
  ordered = NULL,
  dir_prop_cutoff = 0,
  out = NULL,
  sep = NULL,
  header = NULL,
  plot = FALSE,
  sub_feature = "lag & contemp",
  sub_method = "Walktrap",
  sub_sim_thresh = "lowest",
  confirm_subgroup = NULL,
  conv_length = 16,
  conv_interval = 1,
  mean_center_mult = FALSE,
  diagnos = FALSE,
  ms_tol = 1e-05,
  lv_estimator = "miiv",
  lv_scores = "regression",
  lv_miiv_scaling = "first.indicator",
  lv_final_estimator = "miiv"
)
```

Arguments

data	A data.frame in long format with columns for person ID, time-varying variables, and optionally a time/beep column.
vars	Character vector of variable names to model.
id	Character string naming the person-ID column.
time	Character string naming the time/order column, or NULL. When provided, data is sorted by id then time before lagging.
day	Character string naming the day/session column, or NULL. When supplied, lag-1 pairs are formed only within the same (id, day) block, so a lag never crosses the overnight gap.
beep	Character string naming the measurement-occasion column, or NULL. Used (with day) to order observations when time is not given.
min_obs	Integer or NULL. Keep only subjects with at least this many observations (counts taken from data).
subject	Optional vector naming the exact subject(s) to analyse.

ar — Logical. If TRUE (default), autoregressive paths (each variable predicting itself at lag 1) are included as fixed paths.

standardize — Logical. If TRUE (default FALSE), variables are standardized per person before estimation. Note: the returned coefficient network ($coefs, $psi, $temporal_avg, $contemporaneous_avg, $group_paths) is unaffected because Nestimate extracts the standardized lavaan solution (lavInspect(fit, "std")), which is invariant to input scaling. Only the scale-dependent $fit statistics (chisq, aic, bic) change.

groupcutoff — Numeric between 0 and 1. Proportion of individuals for whom a path must be significant to be added at group level. Default 0.75.

subcutoff — Numeric. Subgroup cutoff (default 0.75, matching gimme); only relevant to subgrouping, which is not implemented.

paths — Character vector of lavaan-syntax paths to force into the model (e.g., "V2~V1lag"). Default NULL.

exogenous — Character vector of variable names to treat as exogenous. Default NULL.

hybrid — Logical. If TRUE, also searches residual covariances. Default FALSE.

VAR — Logical. If TRUE, fit a standard VAR: only lagged directed paths are searched and contemporaneous relations are estimated as residual covariances (no directed contemporaneous paths). Matches gimme(VAR = TRUE). Default FALSE.

rmsea_cutoff — Numeric. RMSEA threshold for excellent fit (default 0.05).

srmr_cutoff — Numeric. SRMR threshold for excellent fit (default 0.05).

nnfi_cutoff — Numeric. NNFI/TLI threshold for excellent fit (default 0.95).

cfi_cutoff — Numeric. CFI threshold for excellent fit (default 0.95).

n_excellent — Integer. Number of fit indices that must be excellent to stop individual search. Default 2.

seed — Integer or NULL. Random seed for reproducibility.

group_correct — Character. Group-level multiple-comparison correction; only "Bonferoni Group" (the gimme default) is implemented.

subgroup — Logical. Subgrouping (S-GIMME) is not implemented; TRUE raises an error pointing to gimme::gimme(). Default FALSE.

outcome, conv_vars, mult_vars, lv_model, lasso_model_crit, ms_allow, ordered, dir_prop_cutoff — Accepted for gimme::gimme() API parity but not implemented (latent variable / fMRI-convolution / multiplied-term / LASSO / ordinal / multiple-solutions / directionality features). A non-default value raises an error pointing to gimme::gimme().

out, sep, header, plot — Accepted for gimme::gimme() API parity and ignored: idiographic reads a data.frame (not a CSV directory) and returns an object you plot with plot_gimme(). plot = TRUE emits a message.

sub_feature, sub_method, sub_sim_thresh, confirm_subgroup, conv_length, conv_interval, mean_center_mult, diagnos, ms_tol, lv_estimator, lv_scores, lv_miiv_scaling, lv_final_estimator — Accepted for gimme::gimme() API parity. These configure the unsupported subgrouping / convolution / multiplied-term / multiple-solutions / latent-variable features and are inert here (their parent feature is guarded above).

Value

An S3 object of class "net_gimme" containing:

`temporal` p x p matrix of group-level temporal (lagged) path counts – entry `[i,j]` = number of individuals with path j(t-1)->i(t).

`contemporaneous` p x p matrix of group-level contemporaneous path counts – entry `[i,j]` = number of individuals with path j(t)->i(t).

`coefs` List of per-person p x 2p coefficient matrices (rows = endogenous, cols = `[lagged, contemporaneous]`).

`psi` List of per-person 2p x 2p residual covariance matrices over `c(lag_names, varnames)` (the standardized lavaan psi block), not the p x p current-variable block alone.

`fit` Data frame of per-person fit indices (chisq, df, pvalue, rmsea, srmr, nnfi, cfi, bic, aic, logl, status).

`path_counts` p x 2p matrix: how many individuals have each path.

`paths` List of per-person character vectors of lavaan path syntax.

`group_paths` Character vector of group-level paths found.

`individual_paths` List of per-person character vectors of individual-level paths (beyond group).

`syntax` List of per-person full lavaan syntax strings.

`labels` Character vector of variable names.

`n_subjects` Integer. Number of individuals.

`n_obs` Integer vector. Time points per individual.

`config` List of configuration parameters.

See Also

`build_mlvar`, `graphical_var`, `as_netobject`

Examples

```
# Create simple panel data (3 subjects, 4 variables, 50 time points).
set.seed(42)
n_sub <- 3; n_t <- 50; vars <- paste0("V", 1:4)
rows <- lapply(seq_len(n_sub), function(i) {
  d <- as.data.frame(matrix(rnorm(n_t * 4), ncol = 4))
  names(d) <- vars; d$id <- i; d
})
panel <- do.call(rbind, rows)
res <- build_gimme(panel, vars = vars, id = "id")
print(res)
```

`build_mlvar` *Build a Multilevel Vector Autoregression (mlVAR) network*

Description

Estimates three networks from ESM/EMA panel data, matching `mlVAR::mlVAR()` with `estimator = "lmer"`, `temporal = "fixed"`, `contemporaneous = "fixed"` at machine precision: (1) a directed temporal network of fixed-effect lagged regression coefficients, (2) an undirected contemporaneous network of partial correlations among residuals, and (3) an undirected between-subjects network of partial correlations derived from the person-mean fixed effects.

Usage

```
build_mlvar(
  data,
  vars,
  id,
  day = NULL,
  beep = NULL,
  lags = 1L,
  estimator = c("lmer", "default", "lm", "Mplus"),
  temporal = c("fixed", "correlated", "orthogonal", "unique", "default"),
  contemporaneous = c("fixed", "correlated", "orthogonal", "unique", "default"),
  AR = FALSE,
  scale = FALSE,
  scaleWithin = FALSE,
  nCores = 1L,
  verbose = FALSE,
  lag = NULL,
  standardize = NULL,
  min_obs = NULL,
  subject = NULL
)
```

Arguments

`data`	A `data.frame` containing the panel data.
`vars`	Character vector of variable column names to model.
`id`	Character string naming the person-ID column.
`day`	Character string naming the day/session column, or `NULL`. When provided, lag pairs are only formed within the same day.
`beep`	Character string naming the measurement-occasion column, or `NULL`. When `NULL`, row position within each (id, day) is used.
`lags`	Integer. Lag order; only `1` is supported (mlVAR's `lags`).
`estimator`	Character. Only `"lmer"` / `"default"` are implemented; `"lm"` / `"Mplus"` raise an error (use `mlVAR::mlVAR()`).

`temporal`, `contemporaneous`
: Character. Only `"fixed"` is implemented (idiographic is a clean-room of mlVAR's fixed-effects path). The random-effects modes (`"correlated"`, `"orthogonal"`, `"unique"`) raise an error pointing to `mlVAR::mlVAR()`.

`AR`
: Logical. If `TRUE`, estimate only autoregressive (own-lag) temporal effects, giving a diagonal temporal matrix (matches `mlVAR(AR = TRUE)`). Default `FALSE`.

`scale`
: Logical. If `TRUE`, each variable is grand-mean centered and divided by its pooled SD before augmentation (mlVAR's `scale`). Default `FALSE`. (The deprecated `standardize` is an alias.)

`scaleWithin`
: Logical. If `TRUE`, additionally scale within person (mlVAR's `scaleWithin`). Default `FALSE`.

`nCores`
: Integer. Accepted for API parity; estimation is single-threaded (a message is emitted if `nCores > 1`).

`verbose`
: Logical. Emit progress messages. Default `FALSE`.

`lag`
: Deprecated alias for `lags`.

`standardize`
: Deprecated alias for `scale`.

`min_obs`
: Integer or `NULL`. Keep only subjects with at least this many observations (counts taken from `data`).

`subject`
: Optional vector naming the exact subject(s) to analyse.

Details

The algorithm follows mlVAR's lmer pipeline exactly:

1. Drop rows with NA in id/day/beep and optionally grand-mean standardize each variable.
2. Expand the per-(id, day) beep grid and right-join original values, producing the augmented panel (`augData`).
3. Add within-person lagged predictors (`L1_*`) and person-mean predictors (`PM_*`).
4. For each outcome variable fit `lmer(y ~ within + between-except-own-PM + (1 | id))` with `REML = FALSE`. Collect the fixed-effect temporal matrix `B`, between-effect matrix `Gamma`, random-intercept SDs (`mu_SD`), and lmer residual SDs.
5. Contemporaneous network: `cor2pcor(D %*% cov2cor(cor(resid)) %*% D)`.
6. Between-subjects network: `cor2pcor(pseudoinverse(forcePositive(D (I - Gamma))))`.

Validated to machine precision (max_diff < 1e-10) against `mlVAR::mlVAR()` on 25 real ESM datasets from `openesm` and 20 simulated configurations (seeds 201-220). See `tmp/mlvar_equivalence_real20.R` and `tmp/mlvar_equivalence_20seeds.R`.

Value

A dual-class `c("net_mlvar", "netobject_group")` object — a named list of three full netobjects, one per network, plus model-level metadata stored as attributes. Each element is a standard `c("netobject", "cograph_network")` weight-matrix wrapper (no raw `$data`), so `print()`, `summary()`, `coefs()`, and `cograph::splot(fit$temporal)` work directly. The three constituents are matrix-wrapped and carry no underlying panel data, so any data-resampling workflow (bootstrap, reliability, stability) must start from the original panel rather than from these wrappers. Structure:

`fit$temporal` Directed netobject for the `d x d` matrix of fixed-effect lagged coefficients. `$weights[i, j]` is the effect of variable j at t-lag on variable i at t. `method = "mlvar_temporal"`, `directed = TRUE`.

`fit$contemporaneous` Undirected netobject for the `d x d` partial-correlation network of within-person lmer residuals. `method = "mlvar_contemporaneous"`, `directed = FALSE`.

`fit$between` Undirected netobject for the `d x d` partial-correlation network of person means, derived from `D (I - Gamma)`. `method = "mlvar_between"`, `directed = FALSE`. **Convention:** when a random-intercept SD is 0 the between network is not estimable; idiographic returns an all-zero matrix (with a warning) as a plotting-oriented convention, whereas `mlVAR` returns an all-`NA` matrix. The contemporaneous network follows the same zero-on-degeneracy convention. This is a deliberate departure from strict reference equivalence in the singular case.

`attr(fit, "coefs")` / `coefs()` Tidy `data.frame` with one row per `(outcome, predictor)` pair and columns `outcome`, `predictor`, `beta`, `se`, `t`, `p`, `ci_lower`, `ci_upper`, `significant`. Filter, sort, or plot with base R or the tidyverse. Retrieve with `coefs(fit)`.

`attr(fit, "n_obs")` Number of rows in the augmented panel after na.omit.

`attr(fit, "n_subjects")` Number of unique subjects remaining.

`attr(fit, "lag")` Lag order used.

`attr(fit, "standardize")` Logical; whether pre-augmentation standardization was applied.

See Also

`build_gimme()`, `graphical_var()`, `as_netobject()`

Examples

```
set.seed(1)
n_id <- 8; n_t <- 30; vars <- c("A", "B", "C")
rows <- lapply(seq_len(n_id), function(i) {
  m <- as.data.frame(matrix(rnorm(n_t * 3), ncol = 3))
  names(m) <- vars
  m$id <- i; m$day <- 1L; m$beep <- seq_len(n_t)
  m
})
d <- do.call(rbind, rows)
fit <- build_mlvar(d, vars = vars, id = "id", day = "day", beep = "beep")
print(fit)
summary(fit)
```

`build_mlvar_bayes` *Build a Bayesian multilevel VAR network (Mplus DSEM equivalent)*

Description

Native, pure-R Bayesian estimator for a two-level VAR(1) that statistically reproduces Mplus DSEM output (the estimator behind `mlVAR::mlVAR(estimator = "Mplus")`) without needing Mplus installed. A conjugate Gibbs sampler estimates a fixed temporal matrix, a within-person residual (contemporaneous) network, and a between-person network, using latent mean centering and Mplus's default priors. Point estimates are posterior medians with posterior SDs and 95% credible intervals.

Usage

```
build_mlvar_bayes(
  data,
  vars,
  id,
  day = NULL,
  beep = NULL,
  lags = 1L,
  temporal = c("fixed", "default", "random"),
  contemporaneous = c("fixed", "default"),
  residual = c("fixed", "random"),
  scale = TRUE,
  scaleWithin = FALSE,
  tinterval = NULL,
  impute = FALSE,
  n_iter = 4000L,
  n_burnin = NULL,
  n_chains = 2L,
  thin = 1L,
  seed = NULL,
  min_obs = NULL,
  subject = NULL,
  verbose = FALSE
)
```

Arguments

`data`	A `data.frame` containing the panel data.
`vars`	Character vector of variable column names to model (length >= 2).
`id`	Character string naming the person-ID column.
`day`	Character string naming the day/session column, or `NULL`.
`beep`	Character string naming the measurement-occasion column, or `NULL`. When `NULL`, row position within each (id, day) block is used.
`lags`	Integer lag order; only `1` is supported (matches Mplus DSEM defaults here).
`temporal`	Character. `"fixed"` (default) fits fixed temporal effects with random intercepts (Mplus DSEM `temporal = "fixed"`). `"random"` fits the full DSEM with person-specific temporal matrices `B_i` and a full random-effect covariance over `(mu_i, vec(B_i))`; the temporal network then reports the posterior mean transition matrix and `attr(fit, "slope_sd")` holds the per-coefficient random-slope SDs. `"random"` needs

more subjects estimable random-effect covariance: at least `2 * (p + p^2) + 1` subjects.

`contemporaneous`
: Character. Only `"fixed"` is implemented.

`residual`
: Character. `"fixed"` (default) uses one shared population within-person residual covariance. `"random"` (only with `temporal = "random"`) gives each subject their own residual covariance `Sigma_W_i` via a conjugate hierarchical inverse-Wishart (`Sigma_W_i ~ IW(Lambda, p + 2)`, `Lambda ~ Wishart`), matching DSEM person-specific innovation variances; the reported contemporaneous network is then the population-average residual covariance.

`scale`
: Logical. Global grand-mean/SD standardization of each variable before fitting (Mplus/`mlVAR` `scale = TRUE`). Default `TRUE`.

`scaleWithin`
: Logical. Additionally within-person scale each variable. Default `FALSE`.

`tinterval`
: Numeric or `NULL`. When supplied, `beep` is treated as a continuous time variable and binned onto a regular grid of this width (Mplus `TINTERVAL`); the integer bin becomes the occasion index for gap-aware lagging, and multiple observations in one (id, day, bin) slot are collapsed to the first. Lagging is gap-aware in all cases: lag-1 pairs are only formed between consecutive occasions, so missing occasions never create spurious lag pairs. Default `NULL`.

`impute`
: Logical. If `TRUE` (only with `temporal = "random"`), missing observations are imputed **within the model** each MCMC iteration (data augmentation), rather than dropped: each person's series is expanded to a full occasion grid and every latent cell is drawn from its Gaussian full conditional (as an outcome at `t` and a predictor at `t+1`), using a vectorised even/odd (checkerboard) block sweep. This matches how Mplus / Stan / JAGS handle missing data and removes the listwise-deletion bias under heavy missingness, at extra computational cost. Default `FALSE`.

`n_iter`
: Integer. Total MCMC iterations per chain. Default `4000`.

`n_burnin`
: Integer. Burn-in iterations discarded per chain. Default `n_iter / 2` (Mplus's first-half burn-in convention).

`n_chains`
: Integer. Number of independent chains. Default `2`.

`thin`
: Integer. Keep every `thin`-th post-burn-in draw. Default `1`.

`seed`
: Integer or `NULL`. Base RNG seed (chain `c` uses `seed + c`).

`min_obs`
: Integer or `NULL`. Keep only subjects with at least this many observations before fitting.

`subject`
: Optional vector naming the exact subject(s) to analyse.

`verbose`
: Logical. Emit progress messages. Default `FALSE`.

Details

The sampler alternates five conjugate full-conditional draws per iteration: the latent person means `mu_i` (Gaussian), the fixed temporal matrix `B` (matrix-normal), the within residual covariance `Sigma_W` (inverse-Wishart), the grand mean `alpha` (Gaussian), and the between covariance `Sigma_B` (inverse-Wishart). The lagged predictor is recentred on the current `mu_i` draw every iteration (latent mean

centering). Data are globally standardized first (matching `mlVAR`'s `scale = TRUE`); the first observation of each block is used only as a lag (condition-on-first).

Validated to statistical (Monte-Carlo-error) equivalence against real Mplus 9 DSEM output on standardized synthetic panels: posterior medians of `B`, `Sigma_W`, `Sigma_B` agree with Mplus to well within a posterior SD.

Value

A `net_mlvar_bayes` object (also inheriting `net_mlvar`), a named list of three netobjects (`temporal`, `contemporaneous`, `between`) with posterior-summary attributes. `coefs()` returns a tidy table with `estimate` (posterior median), `posterior_sd`, `ci_lower`, `ci_upper`, `p` (one-tailed), and `significant` (95% CI excludes 0). Posterior draws and the max Gelman-Rubin PSR are kept in attributes.

See Also

`build_mlvar()` (frequentist lmer path), `build_mlvar_mplus()` (true-Mplus wrapper).

Examples

```
set.seed(1)
n_id <- 10; n_t <- 40; vars <- c("A", "B")
rows <- lapply(seq_len(n_id), function(i) {
  y <- matrix(0, n_t, 2)
  for (t in 2:n_t) y[t, ] <- c(0.3, 0.15) * y[t - 1, ] + rnorm(2)
  data.frame(id = i, beep = seq_len(n_t), A = y[, 1], B = y[, 2])
})
d <- do.call(rbind, rows)
fit <- build_mlvar_bayes(d, vars = vars, id = "id", beep = "beep",
                         n_iter = 2000, seed = 1)
print(fit)
coefs(fit)
```

`build_mlvar_mplus` *Build an Mplus-backed multilevel VAR network*

Description

Runs the Mplus Bayesian estimator exposed by `mlVAR::mlVAR(estimator = "Mplus")` and converts the returned posterior summaries into idiographic's network/tidy accessors. This is a true Mplus backend: Mplus must be installed and discoverable by `MplusAutomation::detectMplus()`.

Usage

```
build_mlvar_mplus(
  data,
  vars,
  id,
```

```
  day = NULL,
  beep = NULL,
  lags = 1L,
  temporal = c("fixed", "correlated", "orthogonal", "default"),
  contemporaneous = c("fixed", "correlated", "orthogonal", "default"),
  nCores = 1L,
  scale = TRUE,
  scaleWithin = FALSE,
  MplusSave = TRUE,
  MplusName = "mlVAR_mplus",
  iterations = "(2000)",
  chains = nCores,
  signs,
  min_obs = NULL,
  subject = NULL,
  workdir = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

data	A data.frame containing the panel data.
vars	Character vector of variable column names to model.
id	Character string naming the person-ID column.
day	Character string naming the day/session column, or NULL. Mplus estimation in mlVAR does not directly support day; when supplied it is passed through so mlVAR can prepare the row order, but mlVAR will warn about the Mplus limitation.
beep	Character string naming the measurement-occasion column, or NULL.
lags	Integer lag order. The Mplus backend currently supports 1.
temporal, contemporaneous	Random-effect structure passed to mlVAR. Supported Mplus values are "fixed", "correlated", "orthogonal", and "default".
nCores	Number of Mplus processors/chains.
scale, scaleWithin	Standardization options passed to mlVAR.
MplusSave	Logical. Keep Mplus input/output files in the working directory? Default TRUE.
MplusName	File stem for Mplus input/output files.
iterations	Mplus BITERATIONS string, e.g. "(2000)".
chains	Number of Mplus chains. Defaults to nCores.
signs	Optional sign matrix for contemporaneous random effects.
min_obs	Integer or NULL. Keep only subjects with at least this many observations before fitting.

subject	Optional vector naming the exact subject(s) to analyse.
workdir	Directory in which Mplus files should be written/run. Default uses the current working directory.
verbose	Logical. Show progress from mlVAR/Mplus.
...	Additional arguments passed to mlVAR::mlVAR().

Value

A net_mplus object, also inheriting from net_mlvar, with temporal, contemporaneous, and between networks plus Mplus metadata in attributes. The original mlVAR/Mplus object is available as attr(x, "mplus").

See Also

build_mlvar()

Examples

```
## Not run:
fit <- build_mlvar_mplus(
  data, vars = c("A", "B", "C"), id = "id", beep = "time",
  temporal = "fixed", contemporaneous = "fixed",
  MplusName = "my_mplus_mlvar"
)
edges(fit)
attr(fit, "mplus")$output$summaries

## End(Not run)
```

---

build_usem *Build a user-specified unified SEM network*

Description

Fits person-specific unified Structural Equation Models (uSEM) for intensive longitudinal data. With trim = FALSE, the model is fixed by temporal, contemporaneous, residual_cov, and paths. With trim = TRUE, idiographic uses an independent clean-room modification-index entry and z-value pruning layer over the declared candidate set.

Usage

```
build_usem(
  data,
  vars,
  id,
  time = NULL,
  day = NULL,
```

```
  beep = NULL,
  min_obs = NULL,
  subject = NULL,
  temporal = c("ar", "all", "none"),
  contemporaneous = c("none", "all"),
  residual_cov = TRUE,
  trim = FALSE,
  trim_alpha = 0.05,
  trim_fit_criteria = 3L,
  cfi_cutoff = 0.95,
  tli_cutoff = 0.95,
  rmsea_cutoff = 0.08,
  srmr_cutoff = 0.08,
  paths = NULL,
  exogenous = NULL,
  standardize = FALSE,
  estimator = "ml",
  seed = NULL
)
```

Arguments

`data`	A `data.frame` in long format.
`vars`	Character vector of time-varying variables.
`id`	Character string naming the person-ID column.
`time`	Character string naming the within-person ordering column, or `NULL`.
`day`	Character string naming the day/session column, or `NULL`.
`beep`	Character string naming the measurement-occasion column, or `NULL`.
`min_obs`	Integer or `NULL`. Keep only subjects with at least this many observations.
`subject`	Optional vector naming the subject(s) to analyse.
`temporal`	`"ar"`, `"all"`, `"none"`, or lavaan lagged regressions such as `"A ~ Blag"`.
`contemporaneous`	`"none"`, `"all"`, or lavaan current regressions such as `"B ~ A"`.
`residual_cov`	Logical. Estimate residual covariances among current endogenous variables?
`trim`	Logical. If `TRUE`, use idiographic's clean-room modification-index entry and z-value pruning layer over the declared candidate set.
`trim_alpha`	Significance level used for modification-index entry and z-value pruning when `trim = TRUE`.
`trim_fit_criteria`	Number of fit criteria that must pass before forward search stops.
`cfi_cutoff`, `tli_cutoff`, `rmsea_cutoff`, `srmr_cutoff`	Fit thresholds used by trimmed uSEM.
`paths`	Extra lavaan syntax lines to include unchanged.
`exogenous`	Optional subset of `vars` to treat as exogenous current variables.

standardize	Logical. Standardize variables per person before fitting.
estimator	Lavaan estimator. Default "ml".
seed	Optional random seed.

Value

A net_usem object with average temporal, contemporaneous, and residual-covariance networks, per-subject matrices, fit indices, syntax, and tidy coefficients.

See Also

build_gimme(), graphical_var(), build_mlvar()

---

build_var *Build an ordinary least-squares VAR network*

Description

Fits a transparent VAR(1) baseline from intensive longitudinal data using ordinary least squares. The lag construction, scaling, within-person centering, and day-boundary behavior match graphical_var(), but no regularization or EBIC model selection is applied.

Usage

```
build_var(
  data,
  vars,
  id = NULL,
  day = NULL,
  beep = NULL,
  lags = 1L,
  scale = TRUE,
  center_within = TRUE,
  delete_missings = TRUE,
  min_obs = NULL,
  subject = NULL
)
```

Arguments

data	A data.frame or matrix with columns for variables and optional id/day/beep columns.
vars	Character vector of variable names.
id	Character. Name of the person-ID column, or NULL.
day	Character. Name of the day/session column, or NULL.
beep	Character. Name of the measurement-occasion column, or NULL.

lags	Integer. Only 1 is supported.
scale	Logical. Whether to standardize variables before lagging.
center_within	Logical. Whether to center within person when more than one id is present.
delete_missings	Logical. Drop incomplete current/lagged rows.
min_obs	Integer or NULL. Keep only subjects with at least this many observations.
subject	Optional vector naming the subject(s) to analyse.

Value

A var_result object with temporal OLS coefficients, residual covariance, residual precision, contemporaneous partial correlations, and tidy accessors.

See Also

graphical_var(), build_usem()

---

build_var_bayes *Build a Bayesian VAR(1) network (unregularized Mplus-equivalent)*

Description

Native, pure-R Bayesian VAR(1) that reproduces Mplus's Bayesian (DSEM/time-series) estimates without needing Mplus. It is the unregularized Bayesian counterpart of graphical_var(): instead of a graphical-lasso / EBIC sparse fit, it estimates a full VAR(1) with a flat prior on the temporal coefficients and an inverse-Wishart prior on the residual precision, then reports the temporal network B and the contemporaneous partial-correlation network derived from the residual covariance. With more than one subject the data are within-person centred and pooled (as in graphical_var()).

Usage

```
build_var_bayes(
  data,
  vars,
  id = NULL,
  day = NULL,
  beep = NULL,
  lags = 1L,
  scale = TRUE,
  center_within = TRUE,
  n_iter = 4000L,
  n_burnin = NULL,
  n_chains = 2L,
  thin = 1L,
  seed = NULL,
  min_obs = NULL,
```

```
  subject = NULL,
  verbose = FALSE
)
```

Arguments

data	A data.frame or matrix.
vars	Character vector of variable names (length >= 2).
id	Character. Person-ID column, or NULL for a single series.
day	Character. Day/session column, or NULL.
beep	Character. Beep/measurement column, or NULL.
lags	Integer lag order; only 1 is supported.
scale	Logical. Global standardization of each variable. Default TRUE.
center_within	Logical. Within-person centre when >1 id (removes between-person variance, as in graphical_var()). Default TRUE.
n_iter, n_burnin, n_chains, thin	MCMC controls. Defaults 4000, n_iter/2, 2, 1.
seed	Integer or NULL. Base seed (chain c uses seed + c).
min_obs	Integer or NULL. Keep only subjects with at least this many observations.
subject	Optional vector naming the exact subject(s) to analyse.
verbose	Logical. Progress messages. Default FALSE.

Value

A var_bayes_result object (a cograph group with temporal and contemporaneous netobjects) carrying beta, temporal, kappa, PCC, PDC, posterior draws, and a tidy coefs() table (posterior median, SD, 95% CI, one-tailed p, significance by CI excluding 0).

See Also

graphical_var() (regularized GLASSO/EBIC), build_var() (OLS), build_mlvar_bayes() (multilevel Bayesian VAR).

Examples

```
set.seed(1)
y <- matrix(0, 200, 2)
for (t in 2:200) y[t, ] <- c(0.4, 0.3) * y[t - 1, ] + rnorm(2)
d <- data.frame(A = y[, 1], B = y[, 2])
fit <- build_var_bayes(d, vars = c("A", "B"), n_iter = 2000, seed = 1)
print(fit)
coefs(fit)
```

build_var_each *Fit an ordinary least-squares VAR for every subject*

Description

Applies build_var() to each subject separately, returning one transparent person-specific OLS VAR result per individual. This is the unregularized companion to graphical_var_each() and is useful as an equivalence baseline for checking lag construction, scaling, and temporal coefficient direction.

Usage

```
build_var_each(
  data,
  vars,
  id,
  day = NULL,
  beep = NULL,
  min_obs = NULL,
  ...
)
```

Arguments

data	A data.frame or matrix with columns for variables and optional id/day/beep columns.
vars	Character vector of variable names.
id	Character. Name of the person-ID column; required.
day	Character. Name of the day/session column, or NULL.
beep	Character. Name of the measurement-occasion column, or NULL.
min_obs	Integer or NULL. Keep only subjects with at least this many observations.
...	Further arguments passed to build_var().

Value

A named list of var_result objects (class var_list), one element per subject, named by subject id. Subjects that cannot be fit are dropped with a warning.

See Also

build_var(), graphical_var_each()

Examples

```
set.seed(1)
d <- data.frame(
  id = rep(1:3, each = 40),
  day = rep(1, 120),
  beep = rep(seq_len(40), 3),
  A = rnorm(120), B = rnorm(120), C = rnorm(120)
)
fits <- build_var_each(d, vars = c("A", "B", "C"), id = "id",
                       day = "day", beep = "beep")
fits[["1"]]
```

`coefs` *Tidy coefficients from a fitted mlvar model*

Description

Generic accessor for the tidy coefficient table stored on a `build_mlvar()` result. Returns a `data.frame` with one row per `(outcome, predictor)` pair and columns `outcome`, `predictor`, `beta`, `se`, `t`, `p`, `ci_lower`, `ci_upper`, `significant`.

Usage

```
coefs(x, ...)

## S3 method for class 'net_mlvar'
coefs(x, ...)

## Default S3 method:
coefs(x, ...)

## S3 method for class 'net_mlvar_bayes'
coefs(x, ...)

## S3 method for class 'net_usem'
coefs(x, ...)

## S3 method for class 'var_result'
coefs(x, ...)

## S3 method for class 'var_bayes_result'
coefs(x, ...)

## S3 method for class 'gvar_result'
coefs(x, ...)

## S3 method for class 'net_gimme'
coefs(x, ...)
```

Arguments

x	A fitted model object — currently only net_mlvar is supported.
...	Unused.

Details

Only the within-person (temporal) coefficients are tabulated — these are the lagged fixed effects that populate fit$temporal. The between-subjects effects that go into fit$between are handled via the D (I - Gamma) transformation and are not exposed as a separate tidy table.

Value

A tidy data.frame of coefficient estimates.

Examples

```
set.seed(1)
n_id <- 8; n_t <- 30; vars <- c("A", "B", "C")
rows <- lapply(seq_len(n_id), function(i) {
  m <- as.data.frame(matrix(rnorm(n_t * 3), ncol = 3))
  names(m) <- vars
  m$id <- i; m$day <- 1L; m$beep <- seq_len(n_t)
  m
})
d <- do.call(rbind, rows)
fit <- build_mlvar(d, vars = vars, id = "id", day = "day", beep = "beep")
print(fit)
summary(fit)
```

---

compare_idiographic *Compare idiographic estimators on one dataset*

Description

Fits one or more idiographic estimators to the same data and returns a tidy per-method/per-network comparison table. This is a reporting layer: it does not define a new model, and each row is computed from the estimator's own summary() method plus common edge-table accessors.

Usage

```
compare_idiographic(
  data,
  vars,
  estimators = c("var", "graphical_var"),
  id = NULL,
```

```
  day = NULL,
  beep = NULL,
  estimator_args = list(),
  keep_fits = FALSE
)
```

Arguments

data	A data.frame or matrix with columns for variables and optional id/day/beep columns.
vars	Character vector of variable names.
estimators	Character vector naming estimators to fit. Supported values are "var", "graphical_var", "mlvar", "usem", and "gimme".
id	Character. Name of the person-ID column, or NULL.
day	Character. Name of the day/session column, or NULL.
beep	Character. Name of the measurement-occasion column, or NULL.
estimator_args	Named list of per-estimator argument lists.
keep_fits	Logical. Store fitted model objects?

Value

A model_comparison object with $comparison, $failures, and optionally $fits. $comparison is a tidy data.frame with one row per method/network.

See Also

build_var(), graphical_var(), estimate_stability()

Examples

```
set.seed(1)
d <- data.frame(id = 1, day = rep(1:4, each = 15),
                beep = rep(1:15, 4),
                A = rnorm(60), B = rnorm(60), C = rnorm(60))
cmp <- compare_idiographic(
  d, vars = c("A", "B", "C"), id = "id", day = "day", beep = "beep",
  estimators = c("var", "graphical_var"),
  estimator_args = list(graphical_var = list(n_lambda = 5))
)
cmp$comparison
```

edges.net_usem *Tidy edge table for any idiographic result*

Description

A single tidy verb for every network idiographic produces. Returns one row per edge with columns `network` (e.g. `"temporal"`, `"contemporaneous"`, `"between"`), `from`, `to`, `weight` – and, for GIMME, `level` (`"group"`/`"individual"`). Directed networks (temporal) keep every edge; undirected networks (contemporaneous, between) report each pair once.

Usage

```
## S3 method for class 'net_usem'
edges(x, sort_by = "weight", include_self = FALSE, ...)

## S3 method for class 'var_result'
edges(x, sort_by = "weight", include_self = FALSE, ...)

edges(x, ...)

## S3 method for class 'netobject'
edges(x, sort_by = "weight", include_self = FALSE, ...)

## S3 method for class 'netobject_group'
edges(x, sort_by = "weight", include_self = FALSE, ...)

## S3 method for class 'gvar_result'
edges(x, sort_by = "weight", include_self = FALSE, ...)

## S3 method for class 'net_mlvar'
edges(x, sort_by = "weight", include_self = FALSE, ...)

## S3 method for class 'net_gimme'
edges(
  x,
  sort_by = "weight",
  include_self = TRUE,
  weight = c("prop", "coef"),
  ...
)
```

Arguments

`x`	A `gvar_result`, `net_mlvar`, `net_gimme`, `netobject`, or `netobject_group`.
`sort_by`	`"weight"` (descending |weight|) or `NULL` for natural order.
`include_self`	Keep autoregressive self-loops? Default `FALSE` (`TRUE` for GIMME, where the autoregression is the point).

...	Passed to methods.
weight	For GIMME only: "prop" (proportion of subjects, default) or "coef" (group-average coefficient) for the edge weight.

Value

A tidy data.frame, one row per edge.

Examples

```
set.seed(1)
d <- data.frame(id = 1, A = rnorm(80), B = rnorm(80), C = rnorm(80))
fit <- graphical_var(d, vars = c("A", "B", "C"), id = "id", n_lambda = 8)
edges(fit)            # tidy: network / from / to / weight
```

---

estimate_stability *Estimate edge stability by block resampling (experimental)*

Description

Experimental. The resampling design is methodologically grounded (block bootstrap for dependent data; edge-stability summaries in the spirit of bootnet), but unlike the estimators in this package it has no external reference implementation to validate against, and its interface, defaults, and reported statistics may change in a future release.

Refits an idiographic estimator across deterministic block resamples and summarizes edge-level stability. Blocks preserve within-block time order: subject-day blocks when id and day are supplied, subjects when only id is supplied, days when only day is supplied, or consecutive row blocks for a single series. Duplicate blocks receive temporary ids/day labels before fitting so lag construction never connects two sampled copies.

Usage

```
estimate_stability(
  data,
  vars,
  estimator = c("var", "graphical_var", "mlvar", "usem", "gimme"),
  id = NULL,
  day = NULL,
  beep = NULL,
  n_resamples = 100L,
  resample = c("block", "split_half"),
  block_size = NULL,
  threshold = 1e-08,
  seed = NULL,
```

```
  keep_fits = FALSE,
  ...
)
```

Arguments

data	A data.frame or matrix with columns for variables and optional id/day/beep columns.
vars	Character vector of variable names.
estimator	"var" for build_var(), "graphical_var" for graphical_var(), "mlvar" for build_mlvar(), "usem" for build_usem(), or "gimme" for build_gimme().
id	Character. Name of the person-ID column, or NULL.
day	Character. Name of the day/session column, or NULL.
beep	Character. Name of the measurement-occasion column, or NULL.
n_resamples	Integer number of bootstrap/split resamples.
resample	"block" samples blocks with replacement; "split_half" samples half the blocks without replacement on each replicate.
block_size	Integer or NULL. Consecutive block length used only when neither id nor day is supplied.
threshold	Numeric. Absolute weight above which an edge is counted as selected.
seed	Optional integer seed for deterministic resampling.
keep_fits	Logical. Store successful resampled fits in the returned object?
...	Further arguments passed to the estimator.

Value

A stability_result with $stability edge statistics, $original fit, $resample_edges, $failures, and $config.

See Also

build_var(), graphical_var()

Examples

```
set.seed(1)
d <- data.frame(id = 1, day = rep(1:4, each = 12),
                beep = rep(1:12, 4),
                A = rnorm(48), B = rnorm(48), C = rnorm(48))
st <- estimate_stability(d, vars = c("A", "B", "C"), id = "id",
                         day = "day", beep = "beep",
                         n_resamples = 5, seed = 1)
head(st$stability)
```

extract_edges *Tidy edge table from a network object*

Description

Returns a one-row-per-edge `data.frame` with node labels, for any netobject / cograph_network (or a `gvar_result` constituent).

Usage

```
extract_edges(model, sort_by = "weight", include_self = FALSE)
```

Arguments

`model`	A `netobject` or `cograph_network`. Multi-network results (a `gvar_result`, `net_mlvar`, or any `netobject_group`) hold more than one network, so pass a single constituent — e.g. `extract_edges(as_netobject(x)$temporal)`.
`sort_by`	Either `"weight"` (descending by absolute weight) or `NULL`.
`include_self`	Keep autoregressive self-loops? Default `FALSE`.

Value

A `data.frame` with columns `from`, `to`, `weight`.

graphical_var *Graphical VAR Estimation*

Description

Estimate a graphical vector autoregressive (GVAR) model from time series or panel data. Jointly estimates a sparse temporal network (L1-penalized VAR coefficients) and a sparse contemporaneous network (graphical lasso on residuals) using EBIC model selection over a lambda grid.

Usage

```
graphical_var(
  data,
  vars,
  id = NULL,
  day = NULL,
  beep = NULL,
  lags = 1L,
  n_lambda = 50L,
  gamma = 0.5,
  scale = TRUE,
```

```
  center_within = TRUE,
  lambda_min_ratio = 0.05,
  lambda_min_kappa = NULL,
  lambda_min_beta = NULL,
  penalize_diagonal = TRUE,
  lambda_beta = NULL,
  lambda_kappa = NULL,
  regularize_mat_beta = NULL,
  regularize_mat_kappa = NULL,
  maxit_in = 100L,
  maxit_out = 100L,
  delete_missings = TRUE,
  likelihood = c("unpenalized", "penalized"),
  ebic_tol = 1e-04,
  mimic = "current",
  verbose = FALSE,
  min_obs = NULL,
  subject = NULL
)
```

Arguments

`data`	A data.frame or matrix with columns for variables, and optionally id, day, beep columns for panel/ESM data.
`vars`	Character vector of variable names.
`id`	Character. Name of the person-ID column. If NULL, assumes single subject.
`day`	Character. Name of the day/session column. Default: NULL.
`beep`	Character. Name of the beep/measurement column. Default: NULL.
`lags`	Integer. Lag order. Only `1` is supported (matches `graphicalVAR`'s default; multi-lag is not implemented). Default: 1.
`n_lambda`	Integer. Number of lambda values per penalty dimension. Default: 50 (matches `graphicalVAR`'s `nLambda`).
`gamma`	Numeric. EBIC hyperparameter (0 = BIC, higher = sparser). Default: 0.5.
`scale`	Logical. Whether to standardize variables. Default: TRUE.
`center_within`	Logical. Whether to center within person when more than one id is present (removes between-person variance). Default: TRUE.
`lambda_min_ratio`	Numeric. Ratio of min/max lambda applied to both the beta and kappa grids unless overridden per-dimension. Default: 0.05.
`lambda_min_kappa`, `lambda_min_beta`	Numeric or `NULL`. Per-dimension min/max lambda ratios (matching `graphicalVAR`'s `lambda_min_kappa` / `lambda_min_beta`). When `NULL`, fall back to `lambda_min_ratio`.
`penalize_diagonal`	Logical. Penalize the autoregressive diagonal in beta. Default: TRUE (matches `graphicalVAR`).

`lambda_beta` Numeric scalar (or vector), or `NULL`. When supplied, the temporal penalty is pinned to this value instead of being EBIC-selected over a grid – matching `graphicalVAR`'s `lambda_beta` argument (e.g. `lambda_beta = 0.1`). Default `NULL` (EBIC grid).

`lambda_kappa` Numeric scalar (or vector), or `NULL`. As `lambda_beta` but for the contemporaneous (kappa) penalty.

`regularize_mat_beta`
Optional numeric/logical matrix (`p x p` or `p x (p+1)`) of per-element beta penalty multipliers (matches `graphicalVAR`'s `regularize_mat_beta`). `NULL` uses `penalize_diagonal`.

`regularize_mat_kappa`
Optional `p x p` numeric/logical matrix of per-element kappa penalty multipliers (matches `graphicalVAR`'s `regularize_mat_kappa`). `NULL` penalizes all off-diagonals.

`maxit_in`, `maxit_out`
Integer. Max inner (beta) / outer (beta-kappa) iterations. Defaults 100 (matches `maxit.in` / `maxit.out`).

`delete_missings`
Logical. Drop rows with missing current/lagged values. Default TRUE (matches `deleteMissings`).

`likelihood` Either `"unpenalized"` (default; refit precision for the EBIC, matching `graphicalVAR`) or `"penalized"` (use the regularized kappa directly).

`ebic_tol` Numeric. Tolerance for the EBIC tie-break. Default 1e-4.

`mimic` Character. Only `"current"` is supported (legacy compatibility modes are ignored with a warning).

`verbose` Logical. Emit progress messages. Default FALSE.

`min_obs` Integer or `NULL`. Keep only subjects with at least this many observations (counts taken from `data`). Default `NULL`.

`subject` Optional vector naming the exact subject(s) to analyse. Default `NULL` (all subjects).

Details

This is a clean-room reimplementation of the Rothman/Epskamp two-step estimator that is **numerically equivalent to** `graphicalVAR::graphicalVAR()`: identical data preparation (global scaling, optional within-person centering, intercept column, lag-1 construction within id/day blocks), identical lambda grids (`generate_lambdas`), the coupled MRCE beta-update / glasso kappa-update loop, the unpenalized-likelihood EBIC, and the same tie-broken model selection. On well-conditioned data it agrees with `graphicalVAR` to machine precision (~1e-11); on rank-deficient data (n close to the number of parameters) it agrees to the MRCE inner-solver tolerance (~1e-3), which `graphicalVAR` shares.

Value

A list of class `gvar_result` containing:

beta Temporal coefficient matrix, outcome x (intercept + predictors), in `graphicalVAR`'s convention.

temporal The p x p temporal slice `beta[, -1]` as `[outcome, predictor]` (intercept dropped).

kappa Precision matrix (p x p, symmetric).

PCC Partial contemporaneous correlations `-cov2cor(kappa)`, diagonal zeroed.

PDC Partial directed correlations.

contemporaneous Alias for `PCC`.

labels Variable names.

n_obs Number of valid lag-pair observations.

lambda_beta, lambda_kappa Selected penalties.

gamma, EBIC EBIC gamma used and the selected EBIC.

References

Epskamp, S., Waldorp, L. J., Mottus, R., & Borsboom, D. (2018). The Gaussian Graphical Model in Cross-Sectional and Time-Series Data. *Multivariate Behavioral Research*, 53(4), 453-480.

Rothman, A. J., Levina, E., & Zhu, J. (2010). Sparse multivariate regression with covariance estimation. *JCGS*, 19(4), 947-962.

`graphical_var_each` *Fit a graphical VAR for every subject*

Description

Applies `graphical_var()` to each subject separately, returning one person-specific network per individual — the idiographic "all individuals" workflow. Subjects that cannot be fit (too few lag pairs after listwise deletion) are dropped with a warning.

Usage

```
graphical_var_each(
  data,
  vars,
  id,
  day = NULL,
  beep = NULL,
  min_obs = NULL,
  ...
)
```

Arguments

`data`	A data.frame or matrix with columns for variables, and optionally id, day, beep columns for panel/ESM data.
`vars`	Character vector of variable names.
`id`	Character. The subject-id column (required here).

day	Character. Name of the day/session column. Default: NULL.
beep	Character. Name of the beep/measurement column. Default: NULL.
min_obs	Integer or NULL. Keep only subjects with at least this many observations (counts taken from data). Default NULL.
...	Further arguments passed to graphical_var() (e.g. n_lambda, gamma, scale).

Value

A named list of gvar_result objects (class gvar_list), one element per subject, named by subject id.

---

matrices *Print model matrices for idiographic results*

Description

matrices() is the matrix-oriented companion to summary() and edges(). It returns the core estimated matrices invisibly and prints each matrix compactly with rounding, so users can inspect coefficients without digging through object internals.

Usage

```
matrices(x, ...)

## Default S3 method:
matrices(x, digits = 3, ...)

## S3 method for class 'cograph_network'
matrices(x, digits = 3, ...)

## S3 method for class 'netobject'
matrices(x, digits = 3, ...)

## S3 method for class 'netobject_group'
matrices(x, digits = 3, ...)

## S3 method for class 'gvar_result'
matrices(x, digits = 3, ...)

## S3 method for class 'var_result'
matrices(x, digits = 3, ...)

## S3 method for class 'net_mlvar'
matrices(x, digits = 3, ...)

## S3 method for class 'net_usem'
```

```
matrices(x, digits = 3, ...)

## S3 method for class 'net_gimme'
matrices(x, digits = 3, ...)

## S3 method for class 'preprocess_audit'
matrices(x, digits = 3, ...)

## S3 method for class 'rolling_var_result'
matrices(x, fit = 1L, digits = 3, ...)

## S3 method for class 'rolling_gvar_result'
matrices(x, fit = 1L, digits = 3, ...)

## S3 method for class 'stability_result'
matrices(x, digits = 3, ...)

## S3 method for class 'model_comparison'
matrices(x, fit = 1L, digits = 3, ...)
```

Arguments

x	An idiographic result or cograph network/group.
...	Passed to methods.
digits	Number of digits used for printing. Default 3.
fit	Stored fit name or index for result containers that optionally keep fitted models, such as rolling results and model comparisons.

Value

Invisibly, a named list of matrices.

`nodes.net_usem` *Tidy per-node strength table for any idiographic result*

Description

One row per node per network with `strength` (sum of absolute incident edge weights) and, for directed networks, `out_strength` / `in_strength` (`NA` for undirected). Self-loops are excluded.

Usage

```
## S3 method for class 'net_usem'
nodes(x, ...)

## S3 method for class 'var_result'
```

```
nodes(x, ...)

nodes(x, ...)

## S3 method for class 'netobject'
nodes(x, ...)

## S3 method for class 'netobject_group'
nodes(x, ...)

## S3 method for class 'gvar_result'
nodes(x, ...)

## S3 method for class 'net_mlvar'
nodes(x, ...)

## S3 method for class 'net_gimme'
nodes(x, ...)
```

Arguments

x	A gvar_result, net_mlvar, net_gimme, netobject, or netobject_group.
...	Passed to methods.

Value

A tidy data.frame.

---

plot_gimme *Faithful GIMME network plot (the* gimme*-package convention, via cograph)*

Description

Draws a GIMME result the way the gimme package does: a single p-node network where **dashed edges are lag-1 (temporal)** and **solid edges are lag-0 (contemporaneous)**, **edge width is the proportion of subjects** that have the path, **black edges are group-level** paths and grey edges individual-level, and autoregression shows as a dashed self-loop. Rendered with cograph::splot(), so a lag and a contemporaneous effect between the same pair are drawn as two parallel edges.

Usage

```
plot_gimme(
  x,
  weight = c("prop", "coef"),
  group_color = "black",
  individual_color = "grey60",
```

```
  layout = "circle",
  curvature = 0.25,
  edge_scale = 5,
  ...
)
```

Arguments

x	A net_gimme object from build_gimme().
weight	"prop" (default, proportion of subjects) or "coef" (group-average standardized coefficient) for edge width.
group_color, individual_color	Edge colours for group- vs individual-level paths. Defaults "black" / "grey60".
layout	cograph layout passed to cograph::splot(). Default "circle", matching gimme.
curvature	Edge curvature (separates parallel lag/contemp edges). Default 0.25.
edge_scale	Multiplier mapping weight to drawn line width. Default 5.
...	Further arguments forwarded to cograph::splot().

Value

Invisibly, the mixed cograph_network object that was plotted.

See Also

as_netobject() for the matrix view.

Examples

```
set.seed(1)
panel <- data.frame(
  id = rep(1:5, each = 30),
  t  = rep(seq_len(30), 5),
  A  = rnorm(150), B = rnorm(150), C = rnorm(150)
)
gm <- build_gimme(panel, vars = c("A", "B", "C"), id = "id", time = "t")
plot_gimme(gm)
```

`plot_idiographic` *Plot an idiographic network result*

Description

S3 `plot()` methods that render any idiographic result with `cograph::splot()`. Call `plot(fit)` to draw the full result (every network panel) or pass `layer` to draw a single network – `"temporal"`, `"contemporaneous"`, `"between"` (mlVAR), or `"residual_cov"` (uSEM) – without indexing into the object.

Usage

```
## S3 method for class 'var_result'
plot(x, layer = NULL, ...)

## S3 method for class 'gvar_result'
plot(x, layer = NULL, ...)

## S3 method for class 'var_bayes_result'
plot(x, layer = NULL, ...)

## S3 method for class 'net_mlvar'
plot(x, layer = NULL, ...)

## S3 method for class 'net_usem'
plot(x, layer = NULL, ...)

## S3 method for class 'net_gimme'
plot(x, layer = NULL, weight = c("prop", "coef"), ...)

## S3 method for class 'var_list'
plot(x, subject = 1L, layer = NULL, ...)

## S3 method for class 'gvar_list'
plot(x, subject = 1L, layer = NULL, ...)

## S3 method for class 'rolling_var_result'
plot(x, fit = 1L, layer = NULL, ...)

## S3 method for class 'rolling_gvar_result'
plot(x, fit = 1L, layer = NULL, ...)

## S3 method for class 'stability_result'
plot(x, layer = NULL, ...)
```

Arguments

x	An idiographic result (var_result, gvar_result, net_mlvar, net_usem, net_gimme, var_list, rolling_var_result, rolling_gvar_result, or stability_result).
layer	Optional network name to draw on its own. NULL (default) draws the whole result. Available names are reported if an unknown one is given.
...	Further arguments forwarded to cograph::splot().
weight	For GIMME: "prop" (proportion of subjects, default) or "coef" (group-average coefficient) for edge width.
subject	For a var_list / gvar_list: the subject name (or index) to draw. Defaults to the first subject.
fit	For rolling results: the stored window fit (name or index) to draw. Requires keep_fits = TRUE at fit time. Defaults to the first window.

Value

Invisibly, the object that was plotted (a cograph/ggplot object).

Examples

```
set.seed(1)
d <- data.frame(id = 1, A = rnorm(80), B = rnorm(80), C = rnorm(80))
fit <- build_var(d, vars = c("A", "B", "C"), id = "id")
plot(fit)
plot(fit, layer = "temporal")
```

---

print.forecast_result *Print method for forecast validation results*

Description

Prints a concise summary of estimator, successful rolling splits, prediction count, and overall RMSE.

Usage

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

Arguments

x	A forecast_result object.
...	Ignored.

Value

x, invisibly.

print.gvar_list *Print a list of per-subject graphical VARs*

Description

Print a list of per-subject graphical VARs

Usage

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

Arguments

x	A gvar_list.
...	Unused.

Value

x, invisibly.

print.gvar_result *Print Method for gvar_result*

Description

Print Method for gvar_result

Usage

```
## S3 method for class 'gvar_result'
print(x, digits = 2, ...)
```

Arguments

x	A gvar_result object.
digits	Number of digits used for printed network matrices.
...	Additional arguments (ignored).

Value

The input object, invisibly.

print.model_comparison *Print method for model comparisons*

Description

Prints a concise summary of requested estimators, successful fits, and failures in a model comparison report.

Usage

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

Arguments

x	A model_comparison object.
...	Ignored.

Value

x, invisibly.

print.net_gimme *Print Method for net_gimme*

Description

Print Method for net_gimme

Usage

```
## S3 method for class 'net_gimme'
print(x, digits = 2, ...)
```

Arguments

x	A net_gimme object.
digits	Number of digits used for printed network matrices.
...	Additional arguments (ignored).

Value

The input object, invisibly.

Examples

```
set.seed(1)
panel <- data.frame(
  id = rep(1:5, each = 20),
  t  = rep(seq_len(20), 5),
  A  = rnorm(100), B = rnorm(100), C = rnorm(100)
)
gm <- build_gimme(panel, vars = c("A","B","C"), id = "id", time = "t")
print(gm)
```

`print.net_mlvar` *Print method for net_mlvar*

Description

Print method for net_mlvar

Usage

```
## S3 method for class 'net_mlvar'
print(x, digits = 2, ...)
```

Arguments

`x`	A `net_mlvar` object returned by `build_mlvar()`.
`digits`	Number of digits used for printed network matrices.
`...`	Unused; present for S3 consistency.

Value

Invisibly returns `x`.

Examples

```
set.seed(1)
n_id <- 8; n_t <- 30; vars <- c("A", "B", "C")
rows <- lapply(seq_len(n_id), function(i) {
  m <- as.data.frame(matrix(rnorm(n_t * 3), ncol = 3))
  names(m) <- vars
  m$id <- i; m$day <- 1L; m$beep <- seq_len(n_t)
  m
})
d <- do.call(rbind, rows)
fit <- build_mlvar(d, vars = vars, id = "id", day = "day", beep = "beep")
```

```
print(fit)
summary(fit)
```

`print.net_mlvar_bayes` *Print method for net_mlvar_bayes*

Description

Print method for net_mlvar_bayes

Usage

```
## S3 method for class 'net_mlvar_bayes'
print(x, digits = 2, ...)
```

Arguments

`x`	A `net_mlvar_bayes` object from `build_mlvar_bayes()`.
`digits`	Digits for printed network matrices.
`...`	Unused.

Value

Invisibly returns `x`.

`print.net_usem` *Print method for uSEM fits*

Description

Print method for uSEM fits

Usage

```
## S3 method for class 'net_usem'
print(x, digits = 2, ...)
```

Arguments

`x`	A `net_usem` object.
`digits`	Number of digits used for printed network matrices.
`...`	Ignored.

Value

`x`, invisibly.

print.preprocess_audit

Print method for preprocessing audits

Description

Prints a concise summary of variables, ordered rows, retained lag pairs, trend flags, high-AR flags, and zero-variance flags.

Usage

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

Arguments

x	A preprocess_audit object.
...	Ignored.

Value

x, invisibly.

print.rolling_gvar_result

Print method for rolling graphical VAR results

Description

Prints a concise summary of subject count, window count, and variable count for a rolling graphical VAR result.

Usage

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

Arguments

x	A rolling_gvar_result object.
...	Ignored.

Value

x, invisibly.

print.rolling_var_result

Print method for rolling VAR results

Description

Prints a concise summary of subject count, window count, and variable count for a rolling VAR result.

Usage

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

Arguments

x	A rolling_var_result object.
...	Ignored.

Value

x, invisibly.

print.stability_result

Print method for stability results

Description

Prints a concise summary of estimator, resampling type, successful refits, and the size of the edge-stability table.

Usage

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

Arguments

x	A stability_result object.
...	Ignored.

Value

x, invisibly.

print.var_bayes_result *Print method for var_bayes_result*

Description

Print method for var_bayes_result

Usage

```
## S3 method for class 'var_bayes_result'
print(x, digits = 2, ...)
```

Arguments

x	A var_bayes_result.
digits	Digits for printed networks.
...	Unused.

Value

Invisibly x.

print.var_list *Print a list of per-subject ordinary VARs*

Description

Prints a concise summary of subject count, variable count, lag-pair counts, and temporal edge counts for a build_var_each() result.

Usage

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

Arguments

x	A var_list.
...	Ignored.

Value

x, invisibly.

print.var_result *Print method for ordinary VAR fits*

Description

Print method for ordinary VAR fits

Usage

```
## S3 method for class 'var_result'
print(x, digits = 2, ...)
```

Arguments

x	A var_result object.
digits	Number of digits used for printed network matrices.
...	Ignored.

Value

x, invisibly.

rolling_graphical_var *Estimate rolling-window graphical VAR networks*

Description

Fits graphical_var() over ordered, overlapping windows within each subject. This is the time-varying graphical VAR companion to rolling_var(): every window uses graphical VAR's lag construction, EBIC/penalty settings, and tidy coefficient access, then returns one coefficient table per window.

Usage

```
rolling_graphical_var(
  data,
  vars,
  id = NULL,
  day = NULL,
  beep = NULL,
  window_size,
  step = 1L,
  scale = TRUE,
  center_within = TRUE,
  delete_missings = TRUE,
```

```
  min_obs = NULL,
  subject = NULL,
  keep_fits = FALSE,
  ...
)
```

Arguments

data	A data.frame or matrix with columns for variables and optional id/day/beep columns.
vars	Character vector of variable names.
id	Character. Name of the person-ID column, or NULL.
day	Character. Name of the day/session column, or NULL.
beep	Character. Name of the measurement-occasion column, or NULL.
window_size	Integer number of ordered rows per rolling window.
step	Integer number of rows to advance between windows.
scale	Logical. Whether to standardize variables inside each window.
center_within	Logical. Whether to center within person inside each window when more than one id is present.
delete_missings	Logical. Drop incomplete current/lagged rows.
min_obs	Integer or NULL. Keep only subjects with at least this many observations before rolling.
subject	Optional vector naming the subject(s) to analyse.
keep_fits	Logical. Store successful gvar_result fits?
...	Further arguments passed to graphical_var(), such as n_lambda, gamma, lambda_beta, or lambda_kappa.

Value

A rolling_gvar_result with $estimates, $windows, $failures, and optionally $fits. $estimates is a tidy coefficient table with subject/window metadata plus network, from, to, and weight.

See Also

graphical_var(), rolling_var()

Examples

```
set.seed(1)
d <- data.frame(id = 1, day = rep(1:5, each = 20),
                beep = rep(1:20, 5),
                A = rnorm(100), B = rnorm(100), C = rnorm(100))
tv <- rolling_graphical_var(d, vars = c("A", "B", "C"), id = "id",
                            day = "day", beep = "beep",
                            window_size = 50, step = 25,
                            scale = FALSE, n_lambda = 5)
head(tv$estimates)
```

rolling_var *Estimate rolling-window ordinary VAR networks*

Description

Fits `build_var()` over ordered, overlapping windows within each subject. This is a simple time-varying idiographic baseline: every window uses the same lag construction, scaling, within-person centering, and tidy coefficient access as `build_var()`, but returns one coefficient table per window.

Usage

```
rolling_var(
  data,
  vars,
  id = NULL,
  day = NULL,
  beep = NULL,
  window_size,
  step = 1L,
  scale = TRUE,
  center_within = TRUE,
  delete_missings = TRUE,
  min_obs = NULL,
  subject = NULL,
  keep_fits = FALSE
)
```

Arguments

`data`	A `data.frame` or matrix with columns for variables and optional id/day/beep columns.
`vars`	Character vector of variable names.
`id`	Character. Name of the person-ID column, or `NULL`.
`day`	Character. Name of the day/session column, or `NULL`.
`beep`	Character. Name of the measurement-occasion column, or `NULL`.
`window_size`	Integer number of ordered rows per rolling window.
`step`	Integer number of rows to advance between windows.
`scale`	Logical. Whether to standardize variables inside each window.
`center_within`	Logical. Whether to center within person inside each window when more than one id is present.
`delete_missings`	Logical. Drop incomplete current/lagged rows.
`min_obs`	Integer or `NULL`. Keep only subjects with at least this many observations before rolling.
`subject`	Optional vector naming the subject(s) to analyse.
`keep_fits`	Logical. Store successful `var_result` fits?

Value

A `rolling_var_result` with `$estimates`, `$windows`, `$failures`, and optionally `$fits`. `$estimates` is a tidy coefficient table with subject/window metadata plus `network`, `from`, `to`, and `weight`.

See Also

`build_var()`, `audit_preprocess()`

Examples

```
set.seed(1)
d <- data.frame(id = 1, day = rep(1:5, each = 20),
                beep = rep(1:20, 5),
                A = rnorm(100), B = rnorm(100), C = rnorm(100))
tv <- rolling_var(d, vars = c("A", "B", "C"), id = "id",
                  day = "day", beep = "beep",
                  window_size = 40, step = 20, scale = FALSE)
head(tv$estimates)
```

---

srl *Self-regulated learning intensive longitudinal data (Chapter 20)*

Description

The self-regulated learning (SRL) experience-sampling data used in the Learning Analytics Methods book, Chapter 20 (Vector Autoregression). Each of 36 students reported nine self-regulated-learning indicators once per study occasion for 156 occasions, giving a balanced person-by-time panel suitable for the idiographic time-series methods in this package.

Usage

```
srl
```

Format

A `data.frame` with 5616 rows and 11 columns:

name Student name (36 unique students).

day Within-person occasion index, 1-156.

efficacy Self-efficacy.

value Task value.

planning Planning.

monitoring Monitoring.

effort Effort regulation.

control Control of learning.

help Help seeking.

social Social support.

organizing Organizing.

Details

The columns have already been tidied for modelling: rows are ordered by `name` then `day`, and `day` is a within-person occasion index (1-156) you can pass as the `time` argument to `build_usem()` and `build_gimme()`. No further ordering, indexing, or column selection is needed before fitting a model.

Source

Learning Analytics Methods, Book 2, Chapter 20 (VAR): https://lamethods.org/book2/chapters/ch20-var/ch20-var.html. Original data: https://github.com/lamethods/data2/raw/main/srl/srl.RDS.

Examples

```
data(srl)
summary(srl)
head(srl)
```

`summary.gvar_result` *Summary Method for gvar_result*

Description

Summary Method for gvar_result

Usage

```
## S3 method for class 'gvar_result'
summary(object, ...)
```

Arguments

`object`	A `gvar_result` object.
`...`	Additional arguments (ignored).

Value

A tidy `data.frame` of per-network metrics: one row per network (`temporal`, `contemporaneous`) with `n_nodes`, `n_edges`, `density`, `mean_abs_weight`, `n_positive`, `n_negative`. Use `edges(object)` / `coefs(object)` for the estimates and `nodes(object)` for node strengths.

summary.net_gimme *Summary Method for net_gimme*

Description

Summary Method for net_gimme

Usage

```
## S3 method for class 'net_gimme'
summary(object, ...)
```

Arguments

object	A net_gimme object.
...	Additional arguments (ignored).

Value

A tidy data.frame of per-network metrics (one row per network: temporal, contemporaneous), with n_edges/density/etc. computed from the proportion-of-subjects networks. Per-subject fit indices are in object$fit; coefs(object) gives the per-person estimates, edges(object) the tidy edge list, and nodes(object) node strengths.

Examples

```
set.seed(1)
panel <- data.frame(
  id = rep(1:5, each = 20),
  t  = rep(seq_len(20), 5),
  A  = rnorm(100), B = rnorm(100), C = rnorm(100)
)
gm <- build_gimme(panel, vars = c("A","B","C"), id = "id", time = "t")
summary(gm)
```

summary.net_mlvar *Summary method for net_mlvar*

Description

Summary method for net_mlvar

Usage

```
## S3 method for class 'net_mlvar'
summary(object, ...)
```

Arguments

object	A net_mlvar object returned by build_mlvar().
...	Unused; present for S3 consistency.

Value

A tidy data.frame of per-network metrics (one row per network: temporal, contemporaneous, between). Use coefs(object) for the fixed-effect coefficient table, edges(object) for the edge list, and nodes(object) for node strengths.

Examples

```
set.seed(1)
n_id <- 8; n_t <- 30; vars <- c("A", "B", "C")
rows <- lapply(seq_len(n_id), function(i) {
  m <- as.data.frame(matrix(rnorm(n_t * 3), ncol = 3))
  names(m) <- vars
  m$id <- i; m$day <- 1L; m$beep <- seq_len(n_t)
  m
})
d <- do.call(rbind, rows)
fit <- build_mlvar(d, vars = vars, id = "id", day = "day", beep = "beep")
print(fit)
summary(fit)
```

---

summary.net_usem *Summary method for uSEM fits*

Description

Summary method for uSEM fits

Usage

```
## S3 method for class 'net_usem'
summary(object, ...)
```

Arguments

object	A net_usem object.
...	Ignored.

Value

A tidy per-network metrics `data.frame`.

`summary.var_bayes_result` *Summary method for var_bayes_result*

Description

Summary method for var_bayes_result

Usage

```
## S3 method for class 'var_bayes_result'
summary(object, ...)
```

Arguments

`object`	A `var_bayes_result`.
`...`	Unused.

Value

A tidy per-network metrics `data.frame`.

`summary.var_result` *Summary method for ordinary VAR fits*

Description

Summary method for ordinary VAR fits

Usage

```
## S3 method for class 'var_result'
summary(object, ...)
```

Arguments

`object`	A `var_result` object.
`...`	Ignored.

Value

A tidy per-network metrics `data.frame`.

validate_forecast *Validate one-step forecasts from idiographic VAR models (experimental)*

Description

Experimental. The rolling-origin design follows standard time-series cross-validation practice, but unlike the estimators in this package it has no external reference implementation to validate against, and its interface, defaults, and reported metrics may change in a future release.

Performs rolling-origin one-step prediction from `build_var()` or `graphical_var()`. Each split fits the estimator on earlier blocks and predicts current variables in the next block from their lag-1 values. Scaling and within-person centering parameters are learned from the training split only, then applied to the assessment split before prediction.

Usage

```
validate_forecast(
  data,
  vars,
  estimator = c("var", "graphical_var"),
  id = NULL,
  day = NULL,
  beep = NULL,
  initial = NULL,
  assess = 1L,
  step = 1L,
  n_splits = NULL,
  block_size = NULL,
  scale = TRUE,
  center_within = TRUE,
  delete_missings = TRUE,
  keep_fits = FALSE,
  ...
)
```

Arguments

data	A `data.frame` or matrix with columns for variables and optional id/day/beep columns.
vars	Character vector of variable names.
estimator	`"var"` for `build_var()` or `"graphical_var"` for `graphical_var()`.
id	Character. Name of the person-ID column, or `NULL`.
day	Character. Name of the day/session column, or `NULL`.
beep	Character. Name of the measurement-occasion column, or `NULL`.
initial	Integer number of ordered blocks in the first training split.

assess	Integer number of blocks to assess per split.
step	Integer number of blocks to advance between splits.
n_splits	Optional maximum number of rolling splits.
block_size	Integer or NULL. Consecutive block length used only when neither id nor day is supplied.
scale	Logical. Whether to standardize using training-split means and SDs.
center_within	Logical. Whether to center within person using training-split person means when more than one id is present.
delete_missings	Logical. Drop incomplete current/lagged assessment rows.
keep_fits	Logical. Store fitted split models?
...	Further arguments passed to the estimator.

Value

A forecast_result with $predictions, $metrics, $splits, $failures, and optionally $fits.

See Also

build_var(), graphical_var(), estimate_stability()

Examples

```
set.seed(1)
d <- data.frame(id = 1, day = rep(1:5, each = 12),
                beep = rep(1:12, 5),
                A = rnorm(60), B = rnorm(60), C = rnorm(60))
fc <- validate_forecast(d, vars = c("A", "B", "C"), id = "id",
                        day = "day", beep = "beep",
                        initial = 3, n_splits = 2, scale = FALSE)
fc$metrics
```

Index