Package 'mlVAR'

May 8, 2026

Type Package

Title Multi-Level Vector Autoregression

Version 0.6.1

Depends R (>= 3.3.0)

Imports lme4, arm, qgraph, dplyr (>= 1.0.0), clusterGeneration, mvtnorm, corpcor, abind, methods, parallel, MplusAutomation, graphicalVAR, rlang

Maintainer Sacha Epskamp <mail@sachaepskamp.com>

Description Estimates the multi-level vector autoregression model on time-series data. Three network structures are obtained: temporal networks, contemporaneous networks and between-subjects networks.

License GPL-2

BugReports https://github.com/SachaEpskamp/mlVAR/issues

RoxygenNote 7.3.3

NeedsCompilation no

Author Sacha Epskamp [aut, cre], Marie K. Deserno [aut], Laura F. Bringmann [aut], Myrthe Veenman [ctb], Jonas M. B. Haslbeck [ctb]

Repository CRAN

Date/Publication 2026-03-17 12:50:02 UTC

Contents

`getNet` *Gets a network structure*

Description

This function is simply a wrapper around the plotting method for mlVAR objects, that extracts the network structure rather than plotting them.

Usage

```
getNet(x, ...)
```

Arguments

`x`	An ‘mlVAR‘ or ‘mlVARsim0‘ object.
`...`	Arguments sent to `plot.mlVAR`

Author(s)

Sacha Epskamp <mail@sachaepskamp.com>

`importMplus` *Import output from Mplus*

Description

This function imports the output from an Mplus model that has been generated by mlVAR. It can be used to make manual changes to the input file.

Usage

```
importMplus(outfile)
```

Arguments

`outfile`	Location of Mplus output file.

Author(s)

Sacha Epskamp <mail@sachaepskamp.com>

`mlGGM` *Multi-Level Gaussian Graphical Model Estimation*

Description

Estimates within-cluster and between-cluster partial correlation networks from cross-sectional multilevel data (e.g., individuals nested in classrooms, patients nested in clinics) using a single-step nodewise regression approach. Unlike `mlVAR`, which uses a two-step procedure (temporal model first, then contemporaneous), `mlGGM` fits all effects in a single multilevel model per variable. This makes it suitable for cross-sectional data where no temporal ordering exists.

For each variable y_i, a single mixed-effects model is fitted with within-cluster centered predictors and cluster-mean predictors. Two undirected networks (GGMs) are extracted: one capturing partial correlations at the within-cluster level (individual deviations from cluster means) and one at the between-cluster level (cluster means).

NOTE: This function is currently experimental and may change in future versions.

Usage

```
mlGGM(data, vars, idvar = "id",
      estimator = c("lmer"),
      randomeffects = c("default", "correlated", "orthogonal", "fixed"),
      scale = TRUE, na.rm = TRUE, verbose = TRUE,
      full_detrend = FALSE)

## S3 method for class 'mlGGM'
print(x, ...)
## S3 method for class 'mlGGM'
summary(object, show = c("fit", "within", "between"),
        round = 3, ...)
## S3 method for class 'mlGGM'
plot(x, type = c("within", "between"),
     partial = TRUE, SD = FALSE, subject, order,
     nonsig = c("default", "show", "hide", "dashed"),
     rule = c("or", "and"), alpha = 0.05,
     layout = "spring", verbose = TRUE, ...)
```

Arguments

`data`	A data frame containing the observed variables and a cluster identifier variable.
`vars`	A character vector of variable names to include in the GGM estimation. Must contain at least 2 variables.
`idvar`	A string indicating the name of the cluster/group identifier variable in `data`. Defaults to `"id"`.

estimator	The estimator to be used. Currently only "lmer" (sequential univariate multilevel estimation via lmer) is supported.
randomeffects	How should random effects be estimated? "correlated" estimates correlated random slopes for within-cluster predictors (using (1 + predictors | cluster)), "orthogonal" estimates uncorrelated random slopes (using (1 + predictors || cluster)), and "fixed" estimates only a random intercept with no random slopes. "default" selects "correlated" when the number of variables is 6 or fewer and "orthogonal" otherwise, as correlated random effects become computationally expensive with many variables.
scale	Logical. Should variables be grand-mean standardized before estimation? Defaults to TRUE. Standardization ensures that edge weights are comparable across variables.
na.rm	Logical. Should rows with missing values be removed? Defaults to TRUE.
verbose	Logical indicating if console messages and progress bars should be shown.
full_detrend	Logical. If TRUE, standardize each variable per observation position across all clusters before estimation. For example, all first observations within each cluster are standardized together, all second observations together, etc. Requires a balanced design (equal number of observations per cluster). Applied before grand-mean standardization (scale). Defaults to FALSE.
x	An mlGGM object.
object	An mlGGM object.
show	Character vector indicating which sections to show in the summary. Options: "fit" for information criteria (AIC/BIC), "within" for within-cluster partial correlations and p-values, "between" for between-cluster partial correlations and p-values.
round	Number of decimal places for rounding in the summary output. Defaults to 3.
type	The type of network to plot or extract: "within" for the within-cluster partial correlation network, "between" for the between-cluster partial correlation network.
partial	Logical. If TRUE (default), plots partial correlations. If FALSE, plots zero-order correlations.
SD	Logical. If TRUE, plots random effect standard deviations instead of fixed effects. Only available for the within-cluster network.
subject	Integer specifying which cluster's network to plot. Only available for the within-cluster network. When specified, the cluster-specific network (fixed effects + random effects for that cluster) is plotted.
order	Character vector or numeric vector specifying the order of nodes in the plot.
nonsig	How to handle non-significant edges in the plot: "show" shows all edges regardless of significance, "hide" removes non-significant edges, "dashed" shows non-significant edges as dashed lines. "default" hides non-significant edges for partial correlation networks and shows them otherwise.
rule	Significance rule for the undirected network: "or" requires at least one regression direction to be significant, "and" requires both directions to be significant. The "and" rule is more conservative and tends to improve specificity, especially for the between-cluster network.

alpha	Significance level for thresholding edges. Defaults to 0.05.
layout	Layout algorithm for the network plot, passed to qgraph. Defaults to "spring".
...	Additional arguments passed to qgraph (for plot) or ignored.

Details

mlGGM estimates multi-level Gaussian graphical models by fitting one mixed-effects regression model per variable. For each variable y_i, the model is:

$$y_i = \beta_0 + \sum_{j \neq i} \gamma_{ij}^{(w)} (y_j - \bar{y}_j) + \sum_{j \neq i} \gamma_{ij}^{(b)} \bar{y}_j + u_0 + \sum_{j \neq i} u_j (y_j - \bar{y}_j) + \varepsilon$$

where $(y_j - \bar{y}_j)$ are within-cluster centered predictors and $\bar{y}_j$ are cluster-mean predictors. The random effects u_0, u_j are placed on the intercept and within-cluster predictors only (not on between-cluster means).

Two GGMs are extracted using the relationship between nodewise regression coefficients and the precision matrix:

Within-cluster network: The precision matrix is computed as $K_w = D_w(I - \Gamma_w)$, where $D_w = \mathrm{diag}(1/\sigma_i^2)$ contains the inverse residual variances and Γ_w contains the fixed effects of the within-cluster centered predictors. The matrix is symmetrized and forced to be positive definite. Partial correlations are derived from the precision matrix.

Between-cluster network: The precision matrix is computed as $K_b = D_b(I - \Gamma_b)$, where $D_b = \mathrm{diag}(1/\mu_{SD,i}^2)$ contains the inverse squared random intercept standard deviations and Γ_b contains the fixed effects of the cluster-mean predictors.

Cluster-specific networks: When randomeffects is "correlated" or "orthogonal", cluster-specific within-cluster networks are also computed by adding the estimated random effects to the fixed effects for each cluster. These can be accessed via the subject argument in the plot method.

Network matrices can be extracted using getNet with type = "within" or type = "between". The rule argument controls the significance thresholding rule.

Value

An object of class "mlGGM" containing:

results	A list with: • within: Within-cluster network estimates (covariance, correlation, partial correlation matrices, each with $mean and $subject components) • between: Between-cluster network estimates (covariance, correlation, partial correlation matrices, with $mean component) • Gamma_within: Within-cluster regression coefficients with $mean, $SE, $P, and $subject components • Gamma_between: Between-cluster regression coefficients with $mean, $SE, and $P components • mu: Intercept estimates with $mean, $SE, and $SD components
output	A list of fitted lmer model objects, one per variable.
fit	A data frame with AIC and BIC for each variable's model.

`data`	The augmented data frame used in estimation.
`model`	The predictor model specification data frame.
`input`	A list of input arguments: `vars`, `estimator`, `randomeffects`, `idvar`.

Author(s)

Sacha Epskamp

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.

Epskamp, S., Rhemtulla, M. T., & Borsboom, D. (2017). Generalized network psychometrics: Combining network and latent variable models. *Psychometrika*, 82(4), 904-927.

See Also

`mlVAR` for multilevel VAR estimation with temporal data, `getNet` for extracting network matrices.

Examples

```
## Not run:
# Generate simulated multilevel data with known GGM structure:
library(bootnet)

nNode <- 4
nCluster <- 30
clusterSize <- 50

# True within-cluster and between-cluster networks:
set.seed(1)
net_within <- genGGM(nNode, propPositive = 0.8)
net_between <- genGGM(nNode, p = 1, propPositive = 0.5)

# Generate within-cluster deviations:
within_data <- ggmGenerator()(nCluster * clusterSize, net_within)
within_data <- as.data.frame(within_data)
vars <- names(within_data)
within_data$cluster <- rep(1:nCluster, each = clusterSize)

# Generate between-cluster means:
between_data <- ggmGenerator()(nCluster, net_between)
between_data <- as.data.frame(between_data)
between_data$cluster <- 1:nCluster

# Combine: observed = within deviation + cluster mean
merged <- dplyr::left_join(within_data, between_data, by = "cluster",
                           suffix = c("_w", "_b"))
data <- merged[, grep("_w$", names(merged))] +
        merged[, grep("_b$", names(merged))]
names(data) <- vars
```

```
data$cluster <- merged$cluster

# Fit the multi-level GGM:
fit <- mlGGM(data, vars = vars, idvar = "cluster")

# Inspect results:
summary(fit)

# Plot within-cluster and between-cluster networks:
plot(fit, type = "within")
plot(fit, type = "between")

# Extract network matrices (with AND rule for better specificity):
within_net <- getNet(fit, "within", rule = "and")
between_net <- getNet(fit, "between", rule = "and")

## End(Not run)
```

---

mlVAR *Multilevel VAR Estimation for Multiple Time Series*

Description

The function mlVAR computes estimates of the multivariate vector autoregression model. This model returns three stuctures: temporal effects (e.g., lag-1 regression weights), contemporaneous relationships (correlations or partial correlations) and between-subject effects (correlations and partial correlations). See details.

The predict method computes fitted values and residuals from the temporal model (step 1). It returns predictions aligned to the original input data, with NA for rows that cannot be predicted (e.g., the first observation per person-day lost to lagging, or rows with missing data). When newdata is supplied, predictions are computed for the new data using the fitted model(s). For estimator = "lmer", new subjects are predicted using fixed effects only (allow.new.levels = TRUE); for estimator = "lm", prediction is only possible for subjects seen during training.

The residuals method is a convenience wrapper around predict that returns only the residuals.

The resimulate method generates simulated data from a fitted mlVAR model using person-specific estimated parameters (fixed + random effects). For each person, a VAR time series is simulated using their estimated temporal effects (Beta), contemporaneous precision matrix (Theta), and person-specific means (mu). The output has the same dimensions and missingness pattern as the original input data, making it suitable for posterior predictive checks and model diagnostics.

Usage

```
mlVAR(data, vars, idvar, lags = 1, dayvar, beepvar,
                estimator = c("default", "lmer", "lm","Mplus"),
                contemporaneous = c("default", "correlated",
                "orthogonal", "fixed", "unique"), temporal =
                c("default", "correlated", "orthogonal", "fixed",
```

```
                  "unique"), nCores = 1, verbose = TRUE, compareToLags,
                  scale = TRUE, scaleWithin = FALSE, AR = FALSE,
                  MplusSave = TRUE, MplusName = "mlVAR", iterations = "(2000)",
                  chains = nCores, signs, orthogonal,
                  trueMeans, na.rm = TRUE,
                  full_detrend = FALSE
  )

## S3 method for class 'mlVAR'
predict(object, newdata, scale_back = TRUE,
        include_ids = TRUE, ...)

## S3 method for class 'mlVAR'
residuals(object, scale_back = TRUE,
        include_ids = TRUE, ...)

resimulate(object, ...)

## S3 method for class 'mlVAR'
resimulate(object, scale_back = TRUE,
        include_ids = TRUE, nTime = NULL,
        keep_missing = TRUE,
        variance = c("model", "empirical"), ...)
```

Arguments

data	Data frame
vars	Vectors of variables to include in the analysis
idvar	String indicating the subject ID
lags	Vector indicating the lags to include
dayvar	String indicating assessment day. Adding this argument makes sure that the first measurement of a day is not regressed on the last measurement of the previous day. IMPORTANT: only add this if the data has multiple observations per day.
beepvar	Optional string indicating assessment beep per day. Adding this argument will cause non-consecutive beeps to be treated as missing!
estimator	The estimator to be used. `"lmer"` for sequential univariate multi-level estimation, `"Mplus"` for multivariate Bayesian estimation (requires Mplus), and `"lm"` for fixed effects estimation.
contemporaneous	How should the contemporaneous networks be estimated? These networks are always estimated post-hoc by investigating the residuals of the temporal models. `"correlated"` and `"orthogonal"` run second multi-level models in which the networks are estimated using node-wise estimation. `"fixed"` and `"unique"` simply correlate the residuals, either by computing one network for all subjects (fixed) or a single network per per subject.
temporal	How should the temporal effects be estimated? `"correlated"` estimates correlated random effects, `"orthogonal"` estimates non-correlated random effects

	and "fixed" estimates a model in which only the intercept is random. Defaults to "correlated" when the number of variables is less than 6 and "orthogonal" otherwise. "unique" uses lm to estimate an unique model for each subject.
nCores	Number of cores to use in computation
verbose	Logical indicating if console messages and the progress bar should be shown.
scale	Logical, should variables be standardized before estimation?
scaleWithin	Logial, should variables be scaled within-person (set to FALSE to only center within-person)
compareToLags	A vector indicating which lags to base the data on. If the model is to be compared with a model with multiple lags using mlVARcompare, this argument must be used to make sure the number of observations is the same in both models (e.g., a lag 1 model can model the second observation of a day and a lag-2 model can't, causing different number of observations and incomparable models). It is suggested to not use this argument unless you want to compare models, and always run mlVAR without using this argument afterwards in the selected model.
AR	Logical, should an auto-regression only model be fitted?
MplusSave	Logical, should the Mplus model file and output be saved?
MplusName	Name of the Mplus model file and output (without extensions)
iterations	The string used to define the number of iterations in Mplus
chains	Number of Mplus chains
signs	Optional matrix fixing the signs of contemporaneous correlations. Is estimated by running mlVAR with estimator = "lmer" if missing.
orthogonal	Deprecated argument only added for backward competability. Ignore.
trueMeans	Optional data frame with true person-wise means to use instead of estimated person means. Must contain the idvar column and all vars columns.
na.rm	Logical, should missing values be removed when checking for rank deficiency and when removing incomplete cases? Defaults to TRUE.
full_detrend	Logical. If TRUE, standardize each variable per observation position across all subjects before estimation. For example, all first measurements are standardized together, all second measurements together, etc. This removes systematic occasion effects (e.g., time-of-day trends in ESM data). Requires a balanced design (equal number of observations per subject). Applied before grand-mean standardization (scale). Defaults to FALSE.
object	An mlVAR object (for predict, residuals, and resimulate methods).
newdata	Optional data frame with the same structure as the training data. Must contain the idvar, dayvar, beepvar, and all vars columns. If supplied, predictions are computed for this new data using the fitted model. For estimator = "lmer", new subjects (not seen during training) receive fixed-effects-only predictions. For estimator = "lm", prediction is only available for subjects in the training data.
scale_back	Logical. If TRUE (default) and the model was fitted with scale = TRUE, values are back-transformed to the original scale of the data. If FALSE, values are returned on the standardized (model) scale. Applies to predict, residuals, and resimulate.

include_ids — Logical. If TRUE (default), the idvar, dayvar, and beepvar columns are prepended as columns in the returned data frame. Applies to predict, residuals, and resimulate. When nTime is specified in resimulate, only the idvar column is included.

nTime — Optional integer specifying the number of time points to simulate per person. When NULL (default), the simulated data has the same number of rows as the original input data. When specified, each person gets exactly nTime time points, and the output has nIDs * nTime rows with the idvar and variable columns only (no dayvar or beepvar).

keep_missing — Logical. If TRUE (default) and nTime = NULL, the original variable-level missingness pattern is applied to the simulated data: any cell that was NA in the original data is set to NA in the output. If FALSE, all valid (non-structural-NA) positions receive simulated values. Ignored when nTime is specified.

variance — Character string specifying how the innovation covariance matrix is obtained. "model" (default) uses the model-implied covariance derived from the estimated contemporaneous precision matrix (person-specific GGM structure, population-level residual scale). "empirical" uses the model's partial correlation structure (from the person-specific precision matrix) but scales it by the empirical standard deviations of each person's residuals, i.e., SD %*% cov2cor(solve(kappa_i)) %*% SD. Falls back to "model" for persons with too few residuals.

... — Additional arguments (currently unused).

Details

This function estimates the multi-level VAR model to obtain temporal, contemporaneous and between-subject effects using nodewise estimation. Temporal and between-subject effects are obtained directly from the models and contemporaneous effects are estimated post-hoc by correlating the residuals. See arxiv.org/abs/1609.04156 for details.

Setting estimator = "Mplus" will generate a Mplus model, run the analysis and read the results into R. Mplus 8 is required for this estimation. It is recommended to set contemporaneous = "fixed", though not required. For the estimation of contemporaneous random effects, the signs of contemporaneous *correlations * (not partial correlations) need be set (or estimated) via the signs argument.

Value

mlVAR returns an mlVAR object.

predict.mlVAR returns a data frame of predicted (fitted) values with the same number of rows as the original input data (or newdata). Rows that could not be predicted (e.g., first observation per person-day lost to lagging, or rows with missing data) contain NA. If include_ids = TRUE, the idvar, dayvar, and beepvar columns are prepended.

residuals.mlVAR returns a data frame of residuals with the same structure as predict.mlVAR.

resimulate.mlVAR returns a data frame of simulated values. By default (when nTime = NULL), it has the same number of rows as the original input data, with NAs in the same positions as the original data (unless keep_missing = FALSE). If include_ids = TRUE, the idvar, dayvar, and beepvar columns are prepended. When nTime is specified, the output has nIDs * nTime rows with the idvar

column and variable columns only. Person-specific stationary means are computed from the model parameters, and innovation covariances are derived from the estimated contemporaneous precision matrices.

Author(s)

Sacha Epskamp (mail@sachaepskamp.com)

References

Bringmann, L. F., Vissers, N., Wichers, M., Geschwind, N., Kuppens, P., Peeters, F., ... & Tuerlinckx, F. (2013). A network approach to psychopathology: New insights into clinical longitudinal data. PloS one, 8(4), e60188.

Hamaker, E. L., & Grasman, R. P. (2014). To center or not to center? Investigating inertia with a multilevel autoregressive model. Frontiers in psychology, 5.

Epskamp, S., Waldorp, L. J., Mottus, R., & Borsboom, D. (2017). Discovering Psychological Dynamics: The Gaussian Graphical Model in Cross-sectional and Time-series Data. arxiv.org/abs/1609.04156.

See Also

mlVARcompare, summary.mlVAR, plot.mlVAR, mlGGM

Examples

```
## Not run:
### Small example ###
# Simulate data:
Model <- mlVARsim(nPerson = 50, nNode = 3, nTime = 50, lag=1)

# Estimate using correlated random effects:
fit1 <- mlVAR(Model$Data, vars = Model$vars, idvar = Model$idvar, lags = 1, temporal = "correlated")

# Print some pointers:
print(fit1)

# Summary of all parameter estimates:
summary(fit1)

# Compare temporal relationships:
layout(t(1:2))
plot(Model, "temporal", title = "True temporal relationships", layout = "circle")
plot(fit1, "temporal", title = "Estimated temporal relationships", layout = "circle")

# Compare contemporaneous partial correlations:
layout(t(1:2))
plot(Model, "contemporaneous", title = "True contemporaneous relationships",
    layout = "circle")
plot(fit1, "contemporaneous", title = "Estimated contemporaneous relationships",
    layout = "circle")

# Compare between-subjects partial correlations:
```

```
layout(t(1:2))
plot(Model, "between", title = "True between-subjects relationships", layout = "circle")
plot(fit1, "between", title = "Estimated between-subjects relationships",
    layout = "circle")

# Predictions and residuals (same number of rows as input data):
pred <- predict(fit1)
res <- residuals(fit1)
dim(pred)
dim(res)

# Resimulate data from the fitted model (posterior predictive check):
sim <- resimulate(fit1)
dim(sim)  # same dimensions as original data

# Resimulate with custom time series length (e.g., longer series):
sim_long <- resimulate(fit1, nTime = 500)
dim(sim_long)  # nIDs * 500 rows

# Resimulate without applying original missingness pattern:
sim_complete <- resimulate(fit1, keep_missing = FALSE)
sum(is.na(sim_complete))  # only structural NAs (day/beep boundaries)

# Resimulate with empirical innovation variances (model partial correlations,
# scaled by person-specific empirical residual SDs):
sim_emp <- resimulate(fit1, variance = "empirical")
dim(sim_emp)

# Run same model with non-correlated temporal relationships and fixed-effect model:
fit2 <- mlVAR(Model$Data, vars = Model$vars, idvar = Model$idvar, lags = 1,
    temporal = "orthogonal")
fit3 <- mlVAR(Model$Data, vars = Model$vars, idvar = Model$idvar, lags = 1,
    temporal = "fixed")

# Compare models:
mlVARcompare(fit1,fit2,fit3)

# Inspect true parameter correlation matrix:
Model$model$Omega$cor$mean
# Even though correlations are high, orthogonal model works well often!


### Large example ###
Model <- mlVARsim(nPerson = 100, nNode = 10, nTime = 100,lag=1)

# Correlated random effects no longer practical. Use orthogonal or fixed:
fit4 <- mlVAR(Model$Data, vars = Model$vars, idvar = Model$idvar, lags = 1,
    temporal = "orthogonal")
fit5 <- mlVAR(Model$Data, vars = Model$vars, idvar = Model$idvar, lags = 1,
    temporal = "fixed")

# Compare models:
mlVARcompare(fit4, fit5)
```

```
# Compare temporal relationships:
layout(t(1:2))
plot(Model, "temporal", title = "True temporal relationships", layout = "circle")
plot(fit4, "temporal", title = "Estimated temporal relationships", layout = "circle")

# Compare contemporaneous partial correlations:
layout(t(1:2))
plot(Model, "contemporaneous", title = "True contemporaneous relationships",
    layout = "circle")
plot(fit4, "contemporaneous", title = "Estimated contemporaneous relationships",
    layout = "circle")

# Compare between-subjects partial correlations:
layout(t(1:2))
plot(Model, "between", title = "True between-subjects relationships", layout = "circle")
plot(fit4, "between", title = "Estimated between-subjects relationships",
    layout = "circle")

## End(Not run)
```

`mlVAR-effects` *Fixed and random effects*

Description

These functions return a table of the fixed and random effects.

FUNCTIONS ARE DEPRECATED AND WILL BE REMOVED SOON.

Usage

```
fixedEffects(object, digits = 5)
randomEffects(object, digits = 5)
```

Arguments

`object`	A `mlVAR` object
`digits`	Number of digits to output

Author(s)

Sacha Epskamp (mail@sachaepskamp.com), Marie K. Deserno (m.k.deserno@uva.nl) and Laura F. Bringmann (laura.bringmann@ppw.kuleuven.be)

mlVAR0 *Multilevel VAR Estimation for Multiple Time Series*

Description

The function mlVAR0 computes estimates of the multivariate vector autoregression model as introduced by Bringmann et al. (2013) which can be extended through treatment effects, covariates and pre- and post assessment effects.

FUNCTION IS DEPRECATED AND WILL BE REMOVED SOON.

Usage

```
mlVAR0(data, vars, idvar, lags = 1, dayvar, beepvar,
                periodvar, treatmentvar, covariates, timevar,
                maxTimeDiff, control = list(optimizer = "bobyqa"),
                verbose = TRUE, orthogonal, estimator = c("lmer",
                "lmmlasso"), method = c("default", "stepwise",
                "movingWindow"), laginteractions = c("none", "mains",
                "interactions"), critFun = BIC, lambda = 0,
                center = c("inSubject","general","none"))
```

Arguments

data	Data frame
vars	Vectors of variables to include in the analysis
idvar	String indicating the subject ID
lags	Vector indicating the lags to include
dayvar	String indicating assessment day (if missing, every assessment is set to one day)
beepvar	String indicating assessment beep per day (if missing, is added)
periodvar	String indicating the period (baseline, treatment period, etc.) of assessment (if missing, every assessment is set to one period)
treatmentvar	Character vector indicating treatment
covariates	Character indicating covariates independent of assessment.
timevar	Character indicating the time variable
maxTimeDiff	Maximum time differece to include observation pairs
control	A list of arguments sent to lmerControl
verbose	Logical to print progress to the console
orthogonal	Logical to indicate if orthogonal estimation (no correlated random effects) should be used. Defaults to FALSE if the number of nodes is less than 6 and TRUE otherwise
estimator	Estimator to use. Note: lmmlasso implementation is very experimental
method	Method to use. Experimental

`laginteractions`
: Experimental, do not use.

`critFun`
: Experimental, do not use.

`lambda`
: lmmlasso lambda parameter

`center`
: Centering to be used. `"inSubject"` uses within-person centering, `"general"` uses grand-mean centering and `"none"` does not use centering. IMPORTANT NOTE: `"inSubject"` leads to coefficients to resemble within-person slopes, the other centering option leads to coefficients to be a blend of within and between person slopes.

Details

mlVAR0 has been built to extract individual network dynamics by estimating a multilevel vector autoregression model that models the time dynamics of selected variables both within an individual and on group level. For example, in a lag-1-model each variable at time point t is regressed to a lagged version of itself at time point t-1 and all other variables at time point t-1. In psychological research, for example, this analysis can be used to relate the dynamics of symptoms on one day (as assessed by experience sampling methods) to the dynamics of these symptoms on the consecutive day.

Value

mlVAR0 returns a 'mlVAR0' object containing

`fixedEffects`
: A matrix that contains all fixed effects coefficients with dependent variables as rows and the lagged independent variables as columns.

`se.fixedEffects`
: A matrix that contains all standard errors of the fixed effects.

`randomEffects`
: A list of matrices that contain the random effects coefficients.

`randomEffectsVariance`
: A matrix containing the estimated variances between the random-effects terms

`pvals`
: A matrix that contains p-values for all fixed effects.

`pseudologlik`
: The pseudo log-likelihood.

`BIC`
: Bayesian Information Criterion, i.e. the sum of all univariate models' BICs

`input`
: List containing the names of variables used in the analysis

Author(s)

Sacha Epskamp (mail@sachaepskamp.com), Marie K. Deserno (m.k.deserno@uva.nl) and Laura F. Bringmann (laura.bringmann@ppw.kuleuven.be)

References

Bringmann, L. F., Vissers, N., Wichers, M., Geschwind, N., Kuppens, P., Peeters, F., ... & Tuerlinckx, F. (2013). A network approach to psychopathology: New insights into clinical longitudinal data. PloS one, 8(4), e60188.

See Also

fixedEffects, fixedEffects

Examples

```
## Not run:
### Small network ###
nVar <- 3
nPerson <- 25
nTime <- 25

# Simulate model and data:
Model <- mlVARsim0(nPerson,nVar,nTime,sparsity = 0.5)

# Run mlVAR0:
Res <- mlVAR0(Model)

# Compare true fixed model with significant edges of estimated fixed model:
layout(t(1:2))
plot(Model,"fixed", title = "True model",layout="circle", edge.labels = TRUE)
plot(Res,"fixed", title = "Estimated model", layout = "circle", onlySig = TRUE,
        alpha = 0.05, edge.labels = TRUE)

# Compare true and estimated individual differences in parameters:
layout(t(1:2))
plot(Model,"fixed", title = "True model",layout="circle", edge.color = "blue",
        edge.labels = TRUE)
plot(Res,"fixed", title = "Estimated model", layout = "circle", edge.color = "blue",
        edge.labels = TRUE)

# Compare networks of subject 1:
layout(t(1:2))
plot(Model,"subject",subject = 1, title = "True model",layout="circle",
        edge.labels = TRUE)
plot(Res,"subject",subject = 1,title = "Estimated model", layout = "circle",
        edge.labels = TRUE)



### Large network ###
nVar <- 10
nPerson <- 50
nTime <- 50

# Simulate model and data:
Model <- mlVARsim0(nPerson,nVar,nTime, sparsity = 0.5)

# Run orthogonal mlVAR:
Res <- mlVAR0(Model, orthogonal = TRUE)

# Compare true fixed model with significant edges of estimated fixed model:
layout(t(1:2))
```

```
plot(Model,"fixed", title = "True model",layout="circle")
plot(Res,"fixed", title = "Estimated model", layout = "circle", onlySig = TRUE,
        alpha = 0.05)

# Compare true and estimated individual differences in parameters:
layout(t(1:2))
plot(Model,"fixed", title = "True model",layout="circle", edge.color = "blue")
plot(Res,"fixed", title = "Estimated model", layout = "circle", edge.color = "blue")

# Compare networks of subject 1:
layout(t(1:2))
plot(Model,"subject",subject = 1, title = "True model",layout="circle")
plot(Res,"subject",subject = 1,title = "Estimated model", layout = "circle")

## End(Not run)
```

`mlVAR0-methods` *print and summary functions for mlVAR0 objects*

Description

Create a short summary of an object created by `mlVAR0`.

FUNCTION IS DEPRECATED AND WILL BE REMOVED SOON.

Usage

```
  ## S3 method for class 'mlVAR0'
print(x, ...)
  ## S3 method for class 'mlVAR0'
summary(object, ...)
```

Arguments

`object`	A "mlVAR0" object
`x`	A "mlVAR0" object
`...`	Not used

Author(s)

Sacha Epskamp (mail@sachaepskamp.com), Marie K. Deserno (m.k.deserno@uva.nl) and Laura F. Bringmann (laura.bringmann@ppw.kuleuven.be)

`mlVARcompare` *Compare mlVAR model fit*

Description

This function compares the fit of several mlVAR models. Since an mlVAR model is a combination of univariate models this function will compare the fits for each univariate model.

Usage

```
mlVARcompare(...)
```

Arguments

`...`	Any number of objects obtained from `mlVAR`

Details

Important to note is that the number of observations must be equal to make models comparable. If the lags are different and `compareToLags` was not used in `mlVAR` this function will stop with an informative error message.

Author(s)

Sacha Epskamp (mail@sachaepskamp.com)

Examples

```
## Not run:
### Small example ###
# Simulate data:
Model <- mlVARsim(nPerson = 50, nNode = 3, nTime = 50, lag=1)

# Estimate using different methods:
fit1 <- mlVAR(Model$Data, vars = Model$vars, idvar = Model$idvar, lags = 1,
    temporal = "correlated")
fit2 <- mlVAR(Model$Data, vars = Model$vars, idvar = Model$idvar, lags = 1,
    temporal = "orthogonal")
fit3 <- mlVAR(Model$Data, vars = Model$vars, idvar = Model$idvar, lags = 1,
    temporal = "fixed")

# Compare models:
mlVARcompare(fit1,fit2,fit3)

## End(Not run)
```

`mlVARsample` *Simulator function given an mlVAR object*

Description

Simulates data based on an mlVAR object, estimates the mlVAR network model based on the simulated data and compares the estimated network to the mlVAR object network.

Usage

```
mlVARsample(object, nTime = c(25,50,100,200), nSample = 100, pMissing = 0,
  nReps = 100, nCores = 1, ...)

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

Arguments

`object`	mlVAR object, or mlVARsample object in the summary method
`nTime`	Vector with number of time points to test.
`nSample`	Number of individuals in the dataset. It is possible to decrease the number of individuals compared to the individuals in the mlVAR object. However, it is not possible to have more individuals than there are in the mlVAR object.
`pMissing`	Percentage of missing data to be simulated.
`nReps`	Number of repetitions for each condition.
`nCores`	Number of cores to use.
`...`	Arguments sent to mlVAR.

Details

This function simulates data based on the mlVAR object. The individual networks (random effects) are used to simulate data using the `graphicalVARsim` function from the graphicalVAR package (Epskamp, 2020). The individual data is combined into one dataset. This dataset is used to estimate the mlVAR network.

For every condition, the function returns four values per network comparison measure (correlation, sensitivity, specificity, bias, and precision): one for the fixed temporal effects, one for the fixed contemporaneous effects, the mean comparison value of the random temporal effects, and the mean comparison value of the random contemporaneous effects.

Author(s)

Sacha Epskamp <mail@sachaepskamp.com>

References

Sacha Epskamp (2020). graphicalVAR: Graphical VAR for Experience Sampling Data. R package version 0.2.3. https://CRAN.R-project.org/package=graphicalVAR

See Also

mlVARsim, mlVAR

Examples

```
## Not run:
### Small example ###
# Simulate data:
Model <- mlVARsim(nPerson = 100, nNode = 3, nTime = 50, lag=1)

# Estimate using correlated random effects:
fit <- mlVAR(Model$Data, vars = Model$vars,
             idvar = Model$idvar, lags = 1,
             temporal = "correlated")

# Sample from fitted model:
samples <- mlVARsample(fit, nTime = 50, nSample = 50, pMissing = 0.1,
                       nReps = 5, nCores = 1)

# Summarize results:
summary(samples)

## End(Not run)
```

---

mlVARsim *Simulates an mlVAR model and data*

Description

Simulates an mlVAR model and data with a random variance-covariance matrix for the random effects.

Usage

```
mlVARsim(nPerson = 10, nNode = 5, nTime = 100, lag = 1, thetaVar = rep(1,nNode),
DF_theta = nNode * 2, mu_SD = c(1, 1), init_beta_SD = c(0.1, 1), fixedMuSD = 1,
shrink_fixed = 0.9, shrink_deviation = 0.9, beta_sparsity = 0.5, pcor_sparsity = 0.5)
```

Arguments

nPerson	Number of subjects
nNode	Number of variables
nTime	Number of observations per person

lag	The maximum lag to be used
thetaVar	Contemporaneous fixed effect variances
DF_theta	Degrees of freedom in simulating person-specific contemporaneous covariances (e.g., the individual differences in contemporaneous effects)
mu_SD	Range of standard deviation for the means
init_beta_SD	Initial range of standard deviations for the temporal effects
fixedMuSD	Standard deviation used in sampling the fixed effects
shrink_fixed	Shrinkage factor for shrinking the fixed effects if the VAR model is not stationary
shrink_deviation	Shrinkage factor for shrinking the random effects variance if the VAR model is not stationary
beta_sparsity	Proportion of edges in the temporal network that are set to zero
pcor_sparsity	Proportion of edges in the contemporaneous network that are set to zero

Author(s)

Sacha Epskamp (mail@sachaepskamp.com)

`plot.mlVAR` *Plot Method for mlVAR*

Description

The function `plot.mlVAR` plots estimated model coefficients as networks using `qgraph`. These can be three networks: temporal, contemporaneous and between-subjects effects, of which the latter two can be plotted as a correlation or a partial correlation network.

Usage

```
  ## S3 method for class 'mlVAR'
plot(x, type = c("temporal", "contemporaneous", "between"),
                 lag = 1, partial = TRUE, SD = FALSE, subject, order,
                 nonsig = c("default", "show", "hide", "dashed"), rule
                 = c("or", "and"), alpha = 0.05, onlySig = FALSE,
                 layout = "spring", verbose = TRUE, ...)
  ## S3 method for class 'mlVARsim'
plot(x, ...)
```

Arguments

x	An mlVAR object.
type	What network to plot?
lag	The lag to use when type = "temporal"
partial	Logical, should partial correlation matrices be plotted instead of correlation methods? Only used if type is "contemporaneous" or "between". Defaults to TRUE.
SD	Logical. Plot the standard-deviation of random effects instead of the fixed effect estimate?
subject	Subject number. If not missing, will plot the network of a specific subject instead.
order	An optional character vector used to set the order of nodes in the network.
nonsig	How to handle non-significant edges? Default will hide non-significant edges when p-values are available (fixed effects, partial correlations and temporal effects).
rule	How to choose significance in node-wise estimated GGMs (contemporaneous and between-subjects). "or" selects an edge as being significant if one node predicting the other is significant, and "and" requires both predictions to be significant.
alpha	Alpha level to test for significance
onlySig	Deprecated argument only used for backward competability.
layout	The layout argument used by qgraph
verbose	Logical, should message be printed to the console?
...	Arguments sent to qgraph

Author(s)

Sacha Epskamp (mail@sachaepskamp.com)

---

plot.mlVAR0 *Plot Method for mlVAR0*

Description

The function plot.mlVAR0 plots estimated model coefficients as a network using qgraph.

FUNCTION IS DEPRECATED AND WILL BE REMOVED SOON.

Usage

```
  ## S3 method for class 'mlVAR0'
plot(x, type = c("fixed", "SD", "subject"), lag = 1,
                 subject, order, onlySig = FALSE, alpha, ...)
```

Arguments

x	A mlVAR0 object obtained through the mlVAR0-function
type	Indicates whether to plot a network of fixed effects coefficients ("fixed"), the standard deviations of the random effect terms ("SD") or an individual subject's random effects network ("subject").
lag	Vector indicating the lags to include
subject	If type="subject", vector indicating the ID subject number
order	Order of nodes
onlySig	Logical. Set to TRUE to only plot significant fixed effects.
alpha	Significance level to test edges at if onlySig == TRUE. Defaults to Bonferonni corrected alpha level of 0.05 divided by the number of fixed effects.
...	Arguments sent to qgraph

Author(s)

Sacha Epskamp (mail@sachaepskamp.com), Marie K. Deserno (m.k.deserno@uva.nl) and Laura F. Bringmann (laura.bringmann@ppw.kuleuven.be)

---

simulateVAR *Simulate data from VAR model*

Description

Simulates a timeseries using VAR parameters

Usage

```
simulateVAR(pars,  means = 0, lags = 1, Nt = 100, init, residuals = 0.1,
                burnin)
```

Arguments

pars	A square matrix or a list of square matrices indicating the VAR parameters
means	A vector of means.
lags	The lags to which the 'pars' argument parameters correspond. If 'pars' is a list then this argument should be a vector indicating which lags are represented by each element of the 'pars' list.
Nt	Number of time points
init	Initial setup. Must be a matrix of the first lags with rows corresponding to time points and columns corresponding to variables (e.g., if only two lags are used then the matrix must have two rows indicating the first two times points.)
residuals	Standard deviation of the residuals or a residual covariance matrix
burnin	Initial simulations not returned. Defaults to min(round(Nt/2), 100).

Author(s)

Sacha Epskamp (mail@sachaepskamp.com), Marie K. Deserno (m.k.deserno@uva.nl) and Laura F. Bringmann (laura.bringmann@ppw.kuleuven.be)

`summary.mlVAR` *Summary of mlVAR results*

Description

Prints tables with fit indices and parameter estimates.

Usage

```
## S3 method for class 'mlVAR'
summary(object, show = c("fit", "temporal", "contemporaneous", "between"),
      round = 3, ...)
## S3 method for class 'mlVAR'
print(x, ...)
```

Arguments

`object`	An `mlVAR` object.
`show`	Which tables to show?
`round`	Number of digits.
`x`	An `mlVAR` object.
`...`	Not used

Author(s)

Sacha Epskamp (mail@sachaepskamp.com), Marie K. Deserno (m.k.deserno@uva.nl) and Laura F. Bringmann (laura.bringmann@ppw.kuleuven.be)

Index