Package: dcvar (via r-universe)

June 4, 2026

Title Dynamic Copula VAR Models for Time-Varying Dependence

Version 0.3.1

Description Fits Bayesian copula vector autoregressive models for bivariate time series with dynamic, regime-switching, and constant dependence structures. The package includes simulation, data preparation, estimation with 'Stan' through 'rstan' or 'cmdstanr', posterior summaries, diagnostics, trajectory extraction, fitted and predictive summaries, and approximate leave-one-out cross-validation model comparison for supported fits. For Bayesian computation and model comparison, see Carpenter et al. (2017) <doi:10.18637/jss.v076.i01> and Vehtari, Gelman and Gabry (2017) <doi:10.1007/s11222-016-9696-4>.

License GPL (>= 3)

URL `https://github.com/benlug/dcvar`

BugReports `https://github.com/benlug/dcvar/issues`

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports rstan (>= 2.26.0), posterior (>= 1.5.0), loo (>= 2.7.0), ggplot2 (>= 3.4.0), patchwork (>= 1.1.0), bayesplot (>= 1.10.0), rlang (>= 1.0.0), cli (>= 3.0.0), parallel, stats, tools, utils

Suggests cmdstanr (>= 0.8.0), testthat (>= 3.0.0), knitr, rmarkdown, withr, sn

Additional_repositories `https://stan-dev.r-universe.dev`

VignetteBuilder knitr

Config/testthat/edition 3

LazyData false

Config/pak/sysreqs make libicu-dev

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

Date/Publication 2026-06-04 14:41:35 UTC

RemoteUrl https://github.com/benlug/dcvar

RemoteRef HEAD

RemoteSha 28910395dbda7ad7ac94195d2d2efa4fa5e0ee41

Contents

`aggregate_metrics` *Aggregate metrics across simulation replications*

Description

Aggregate metrics across simulation replications

Usage

```
aggregate_metrics(metrics_list)
```

Arguments

`metrics_list`	A list of metric lists (one per replication), each containing a `$rho` element as returned by `compute_rho_metrics()`.

Value

A named list with:

- `rho`: data frame of aggregated rho metrics (mean, SD, quantiles) for every field in the per-replication `rho` metric list
- `n_reps`: number of replications

`as.data.frame.dcvar_model_fit`

Convert a dcvar fit summary to a data frame

Description

Returns the full parameter summary as a tidy data frame with correct 2.5%/97.5% quantiles.

Usage

```
## S3 method for class 'dcvar_model_fit'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	A fitted dcvar model object (any subclass of `dcvar_model_fit`: `dcvar_fit`, `dcvar_constant_fit`, `dcvar_hmm_fit`, `dcvar_covariate_fit`, `dcvar_sem_fit`, or `dcvar_multilevel_fit`).
`row.names`	Ignored.
`optional`	Ignored.
`...`	Additional arguments (unused).

Value

A data frame with columns `variable`, `mean`, `sd`, `q2.5`, `q97.5`, `rhat`, `ess_bulk`, `ess_tail`.

`compute_param_metrics`

Compute scalar parameter recovery metrics

Description

Compute scalar parameter recovery metrics

Usage

```
compute_param_metrics(true_value, est_mean, est_lower, est_upper)
```

Arguments

`true_value`	True parameter value.
`est_mean`	Estimated posterior mean.
`est_lower`	Lower interval bound (2.5% quantile).
`est_upper`	Upper interval bound (97.5% quantile).

Value

A named list with `bias`, `relative_bias`, `covered`, `interval_width`.

`compute_rho_metrics` *Compute rho trajectory recovery metrics*

Description

Evaluates how well the estimated rho trajectory recovers the true values.

Usage

```
compute_rho_metrics(rho_true, rho_est, rho_lower, rho_upper)
```

Arguments

`rho_true`	Numeric vector of true rho values (length T-1).
`rho_est`	Numeric vector of estimated rho (posterior mean).
`rho_lower`	Numeric vector of lower interval bounds (2.5% quantiles).
`rho_upper`	Numeric vector of upper interval bounds (97.5% quantiles).

Value

A named list with:

- `bias`: mean bias
- `relative_bias`: mean relative bias (%)
- `coverage`: proportion of intervals containing true value
- `interval_width`: mean interval width
- `correlation`: Pearson correlation between true and estimated
- `bias_start`, `bias_end`: bias at first/last time point
- `coverage_start`, `coverage_end`: coverage at endpoints

`covariate_effects` *Extract covariate effect summaries*

Description

Returns posterior summaries for the Fisher-z intercept and covariate effects. The residual random-walk innovation scale `sigma_omega` is reported separately by `var_params()` for `drift` = `TRUE` fits.

Usage

```
covariate_effects(object, ...)

## Default S3 method:
covariate_effects(object, ...)

## S3 method for class 'dcvar_covariate_fit'
covariate_effects(object, probs = c(0.025, 0.5, 0.975), ...)
```

Arguments

`object`	A `dcvar_covariate_fit` object.
`...`	Additional arguments (unused).
`probs`	Numeric vector of quantile probabilities (default: `c(0.025, 0.5, 0.975)`).

Value

A data frame with one row per effect and columns `term`, `variable`, `mean`, `sd`, and one column per requested quantile.

`dcvar` *Fit the DC-VAR model*

Description

Fits a Dynamic Copula VAR(1) model with time-varying correlation following a random walk on the Fisher-z scale. Uses non-centered parameterisation for efficient HMC sampling.

Usage

```
dcvar(
  data,
  vars,
  time_var = "time",
  standardize = TRUE,
  margins = "normal",
```

```
  skew_direction = NULL,
  allow_gaps = FALSE,
  prior_mu_sd = 2,
  prior_phi_sd = 0.5,
  prior_sigma_eps_rate = 1,
  prior_sigma_omega_rate = 0.1,
  prior_rho_init_sd = 1,
  chains = 4,
  iter_warmup = 2000,
  iter_sampling = 4000,
  adapt_delta = 0.99,
  max_treedepth = 12,
  seed = NULL,
  cores = NULL,
  refresh = 500,
  init = NULL,
  stan_file = NULL,
  backend = getOption("dcvar.backend", "auto"),
  ...
)
```

Arguments

`data`	A data frame with time series observations.
`vars`	Character vector of two variable names to model.
`time_var`	Name of the time column (default: `"time"`).
`standardize`	Logical; whether to z-score variables (default: `TRUE`).
`margins`	Marginal distribution specification. Either a single string applied to both variables, or a length-2 character vector giving a per-variable (mixed) margin, e.g. `c("normal", "exponential")`. Each entry is one of `"normal"` (default), `"exponential"`, `"skew_normal"`, or `"gamma"`. When the two entries differ the fit uses a generic mixed-margins Stan model under the Gaussian copula; identical entries reuse the specialised single-family model.
`skew_direction`	Integer vector of length 2 indicating skew direction for asymmetric margins. Each element must be `1` (right-skewed) or `-1` (left-skewed). Required whenever any dimension uses an `"exponential"` or `"gamma"` margin; only those dimensions consult it.
`allow_gaps`	Logical; if `FALSE` (default), interior missing values cause an error because they break VAR(1) time series adjacency. Set to `TRUE` to allow fitting with a warning instead.
`prior_mu_sd`	Prior SD for intercepts: `mu ~ normal(0, prior_mu_sd)`.
`prior_phi_sd`	Prior SD for VAR coefficients: `Phi ~ normal(0, prior_phi_sd)`.
`prior_sigma_eps_rate`	Prior mean for innovation SDs (see `prepare_dcvar_data()`).
`prior_sigma_omega_rate`	Prior mean for rho process SD (see `prepare_dcvar_data()`).

`prior_rho_init_sd`	Prior SD for initial rho on Fisher-z scale.
`chains`	Number of MCMC chains (default: 4).
`iter_warmup`	Warmup iterations per chain (default: 2000).
`iter_sampling`	Sampling iterations per chain (default: 4000).
`adapt_delta`	Target acceptance rate (default: 0.99). The DC-VAR model uses a lower default than `dcvar_constant()` (0.999) because the non-centered parameterisation already handles posterior geometry well.
`max_treedepth`	Maximum tree depth (default: 12).
`seed`	Random seed.
`cores`	Number of parallel chains. `NULL` uses all available cores.
`refresh`	How often to print progress (default: 500). Set to 0 for silent operation.
`init`	Custom init function or `NULL` for smart defaults.
`stan_file`	Path to a custom Stan file, or `NULL` to use the bundled model.
`backend`	Character: `"auto"` (default, uses rstan), `"rstan"`, or `"cmdstanr"`. Can also be set globally via `options(dcvar.backend = "cmdstanr")`.
`...`	Additional backend-specific sampling arguments.

Value

A `dcvar_fit` object.

See Also

`dcvar_constant()` for the time-invariant baseline, `dcvar_hmm()` for the regime-switching model, `dcvar_compare()` for LOO-CV model comparison, `rho_trajectory()` and `plot_rho()` for inspecting results.

Examples

```
sim <- simulate_dcvar(
  n_time = 12,
  rho_trajectory = rho_decreasing(12),
  seed = 1
)
fit <- dcvar(
  sim$Y_df,
  vars = c("y1", "y2"),
  chains = 1,
  iter_warmup = 10,
  iter_sampling = 10,
  refresh = 0,
  seed = 1
)
print(fit)
summary(fit)
plot(fit)
```

`dcvar_compare` *Compare multiple fitted models using LOO-CV*

Description

Convenience wrapper around `loo::loo_compare()` that accepts named dcvar model fits.

Usage

```
dcvar_compare(...)
```

Arguments

`...`	Named fitted model objects (e.g., `dcvar = fit1, hmm = fit2`).

Value

A `loo_compare` matrix.

See Also

`loo::loo_compare()` for details on the comparison method, `dcvar()`, `dcvar_hmm()`, `dcvar_constant()` for fitting models.

Examples

```
sim <- simulate_dcvar(
  n_time = 12,
  rho_trajectory = rho_decreasing(12),
  seed = 1
)
fit_dcvar <- dcvar(
  sim$Y_df,
  vars = c("y1", "y2"),
  chains = 1,
  iter_warmup = 10,
  iter_sampling = 10,
  refresh = 0,
  seed = 1
)
fit_constant <- dcvar_constant(
  sim$Y_df,
  vars = c("y1", "y2"),
  chains = 1,
  iter_warmup = 10,
  iter_sampling = 10,
  refresh = 0,
  seed = 1
)
```

```
dcvar_compare(dcvar = fit_dcvar, constant = fit_constant)
```

`dcvar_constant` *Fit the constant copula model*

Description

Fits a copula VAR(1) model with a single time-invariant correlation parameter. This serves as a baseline for comparison with the DC-VAR and HMM copula models.

Usage

```
dcvar_constant(
  data,
  vars,
  time_var = "time",
  standardize = TRUE,
  margins = "normal",
  copula = "gaussian",
  skew_direction = NULL,
  allow_gaps = FALSE,
  prior_mu_sd = 2,
  prior_phi_sd = 0.5,
  prior_sigma_eps_rate = 1,
  prior_z_rho_sd = 1,
  chains = 4,
  iter_warmup = 2000,
  iter_sampling = 4000,
  adapt_delta = 0.999,
  max_treedepth = 12,
  seed = NULL,
  cores = NULL,
  refresh = 500,
  init = NULL,
  stan_file = NULL,
  backend = getOption("dcvar.backend", "auto"),
  ...
)
```

Arguments

`data`	A data frame with time series observations.
`vars`	Character vector of two variable names to model.
`time_var`	Name of the time column (default: `"time"`).
`standardize`	Logical; whether to z-score variables (default: `TRUE`).

`margins`	Marginal distribution specification. Either a single string applied to both variables, or a length-2 character vector giving a per-variable (mixed) margin (for example `c("normal", "exponential")`). Each entry is one of `"normal"` (default), `"exponential"`, `"skew_normal"`, or `"gamma"`. When the two entries differ the fit uses a generic mixed-margins Stan model; identical entries are equivalent to the scalar form and reuse the specialised single-family model. Mixed margins are supported with the Gaussian copula, and (for `dcvar_constant`) additionally with the Clayton copula.
`copula`	Character string specifying the copula family. One of `"gaussian"` (default) or `"clayton"`. Clayton is currently available with normal margins or a per-variable (mixed) margin vector; a single non-normal family with Clayton is not yet supported.
`skew_direction`	Integer vector of length 2 indicating skew direction for asymmetric margins. Each element must be `1` (right-skewed) or `-1` (left-skewed). Required whenever any dimension uses an `"exponential"` or `"gamma"` margin; only those dimensions consult it.
`allow_gaps`	Logical; if `FALSE` (default), interior missing values cause an error because they break VAR(1) time series adjacency. Set to `TRUE` to allow fitting with a warning instead.
`prior_mu_sd`	Prior SD for intercepts: `mu` ~ `normal(0, prior_mu_sd)`.
`prior_phi_sd`	Prior SD for VAR coefficients: `Phi` ~ `normal(0, prior_phi_sd)`.
`prior_sigma_eps_rate`	Prior mean for innovation SDs (see `prepare_dcvar_data()`).
`prior_z_rho_sd`	Prior SD for rho on Fisher-z scale (default: 1.0).
`chains`	Number of MCMC chains (default: 4).
`iter_warmup`	Warmup iterations per chain (default: 2000).
`iter_sampling`	Sampling iterations per chain (default: 4000).
`adapt_delta`	Target acceptance rate (default: 0.999). The constant model uses a higher default than DC-VAR (0.99) because its simpler posterior geometry benefits from tighter step-size adaptation without significant cost, reducing occasional divergences near the rho boundary.
`max_treedepth`	Maximum tree depth (default: 12).
`seed`	Random seed.
`cores`	Number of parallel chains. `NULL` uses all available cores.
`refresh`	How often to print progress (default: 500). Set to 0 for silent operation.
`init`	Custom init function or `NULL` for smart defaults.
`stan_file`	Path to a custom Stan file, or `NULL` to use the bundled model.
`backend`	Character: `"auto"` (default, uses rstan), `"rstan"`, or `"cmdstanr"`. Can also be set globally via `options(dcvar.backend = "cmdstanr")`.
`...`	Additional backend-specific sampling arguments.

Details

`adapt_delta` defaults to 0.999 because the constant-rho model has a simpler correlation structure that benefits from tighter step-size adaptation without significant computational cost, reducing occasional divergences near the rho boundary.

Value

A `dcvar_constant_fit` object.

See Also

`dcvar()` for the time-varying model, `dcvar_hmm()` for the regime-switching model, `dcvar_compare()` for LOO-CV model comparison.

Examples

```
sim <- simulate_dcvar(
  n_time = 12,
  rho_trajectory = rho_constant(12, rho = 0.5),
  seed = 1
)
fit <- dcvar_constant(
  sim$Y_df,
  vars = c("y1", "y2"),
  chains = 1,
  iter_warmup = 10,
  iter_sampling = 10,
  refresh = 0,
  seed = 1
)
print(fit)
```

---

dcvar_constant_fit-methods

S3 methods for dcvar_constant_fit objects

Description

S3 methods for dcvar_constant_fit objects

Usage

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

## S3 method for class 'dcvar_constant_fit'
summary(object, probs = c(0.025, 0.5, 0.975), ...)
```

```
## S3 method for class 'dcvar_constant_fit'
coef(object, ...)

## S3 method for class 'dcvar_constant_fit'
plot(x, type = c("rho", "phi", "diagnostics", "ppc", "pit"), ...)
```

Arguments

`x, object`	A `dcvar_constant_fit` object.
`...`	Additional arguments (unused).
`probs`	Numeric vector of quantile probabilities for the rho estimate (default: `c(0.025, 0.5, 0.975)`).
`type`	Character; one of `"rho"`, `"phi"`, `"diagnostics"`, `"ppc"`, `"pit"`.

Value

Invisibly returns `x`.

A `dcvar_constant_summary` object (a list).

A named list of posterior means: `mu`, `Phi`, margin-specific scale/shape parameters (e.g. `sigma_eps` for normal margins), and either `rho` (Gaussian copula) or `theta` (Clayton copula).

A ggplot object.

Functions

- `print(dcvar_constant_fit)`: Print a concise overview of the constant copula fit.
- `summary(dcvar_constant_fit)`: Produce a detailed summary including constant rho, VAR parameters, and diagnostics.
- `coef(dcvar_constant_fit)`: Extract posterior means of model coefficients.
- `plot(dcvar_constant_fit)`: Dispatch to a plot type: `"rho"`, `"phi"`, `"diagnostics"`, `"ppc"`, or `"pit"`.

`dcvar_covariate` *Fit the covariate DC-VAR model*

Description

Fits a Gaussian Dynamic Copula VAR(1) model in which contemporaneous innovation dependence varies on the Fisher-z scale as a function of observed covariates. With `drift = TRUE`, the model adds residual random-walk drift: `zeta_i = beta_0 + x_{i+1}' beta + eta_i`. With `drift = FALSE`, the dependence trajectory is explained entirely by the covariates.

Usage

```
dcvar_covariate(
  data,
  vars,
  covariates,
  time_var = "time",
  standardize = TRUE,
  standardize_covariates = FALSE,
  drift = TRUE,
  zero_init_eta = TRUE,
  allow_gaps = FALSE,
  prior_mu_sd = 2,
  prior_phi_sd = 0.5,
  prior_sigma_eps_rate = 1,
  prior_sigma_omega_rate = 0.1,
  prior_rho_init_sd = 1,
  prior_beta_sd = 1,
  chains = 4,
  iter_warmup = 2000,
  iter_sampling = 4000,
  adapt_delta = 0.99,
  max_treedepth = 12,
  seed = NULL,
  cores = NULL,
  refresh = 500,
  init = NULL,
  stan_file = NULL,
  backend = getOption("dcvar.backend", "auto"),
  ...
)
```

Arguments

`data`	A data frame with time series observations.
`vars`	Character vector of two variable names to model.
`covariates`	Character vector of covariate column names.
`time_var`	Name of the time column (default: `"time"`).
`standardize`	Logical; whether to z-score variables (default: `TRUE`).
`standardize_covariates`	Logical; whether to z-score covariates (default: `FALSE`).
`drift`	Logical; if `TRUE` (default), include residual random-walk drift after the covariate effect. If `FALSE`, fit the no-drift sensitivity model.
`zero_init_eta`	Logical; if `TRUE` (default), fixes `eta[1] = 0` in the residual-drift model.
`allow_gaps`	Logical; if `FALSE` (default), interior missing values cause an error because they break VAR(1) time series adjacency. Set to `TRUE` to allow fitting with a warning instead.

`prior_mu_sd`	Prior SD for intercepts: `mu` ~ `normal(0, prior_mu_sd)`.
`prior_phi_sd`	Prior SD for VAR coefficients: `Phi` ~ `normal(0, prior_phi_sd)`.
`prior_sigma_eps_rate`	Prior mean for innovation SDs (see `prepare_dcvar_data()`).
`prior_sigma_omega_rate`	Prior mean for rho process SD (see `prepare_dcvar_data()`).
`prior_rho_init_sd`	Prior SD for initial rho on Fisher-z scale.
`prior_beta_sd`	Prior SD for covariate effects.
`chains`	Number of MCMC chains (default: 4).
`iter_warmup`	Warmup iterations per chain (default: 2000).
`iter_sampling`	Sampling iterations per chain (default: 4000).
`adapt_delta`	Target acceptance rate (default: 0.99). The DC-VAR model uses a lower default than `dcvar_constant()` (0.999) because the non-centered parameterisation already handles posterior geometry well.
`max_treedepth`	Maximum tree depth (default: 12).
`seed`	Random seed.
`cores`	Number of parallel chains. `NULL` uses all available cores.
`refresh`	How often to print progress (default: 500). Set to 0 for silent operation.
`init`	Custom init function or `NULL` for smart defaults.
`stan_file`	Path to a custom Stan file, or `NULL` to use the bundled model.
`backend`	Character: `"auto"` (default, uses rstan), `"rstan"`, or `"cmdstanr"`. Can also be set globally via `options(dcvar.backend = "cmdstanr")`.
`...`	Additional backend-specific sampling arguments.

Value

A `dcvar_covariate_fit` object.

See Also

`prepare_dcvar_covariate_data()`, `covariate_effects()`, `rho_trajectory()`, and `dcvar()` for the covariate-free random-walk model.

Examples

```
sim <- simulate_dcvar(
  n_time = 12,
  rho_trajectory = rho_step(12),
  seed = 1
)
sim$Y_df$phase <- as.numeric(sim$Y_df$time > 6)
fit <- dcvar_covariate(
  sim$Y_df,
  vars = c("y1", "y2"),
```

```
  covariates = "phase",
  chains = 1,
  iter_warmup = 10,
  iter_sampling = 10,
  refresh = 0,
  seed = 1
)
covariate_effects(fit)

fit_nodrift <- dcvar_covariate(
  sim$Y_df,
  vars = c("y1", "y2"),
  covariates = "phase",
  drift = FALSE,
  chains = 1,
  iter_warmup = 10,
  iter_sampling = 10,
  refresh = 0,
  seed = 1
)
rho_trajectory(fit_nodrift)
```

---

dcvar_covariate_fit-methods

S3 methods for covariate DC-VAR fits

Description

S3 methods for covariate DC-VAR fits

Usage

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

## S3 method for class 'dcvar_covariate_fit'
summary(object, probs = c(0.025, 0.5, 0.975), ...)

## S3 method for class 'dcvar_covariate_fit'
coef(object, ...)

## S3 method for class 'dcvar_covariate_fit'
plot(x, type = c("rho", "phi", "diagnostics", "ppc", "pit"), ...)
```

Arguments

`x, object`	A `dcvar_covariate_fit` object.
`...`	Additional arguments (unused).

`probs`	Numeric vector of quantile probabilities (default: `c(0.025, 0.5, 0.975)`).
`type`	Character; one of `"rho"`, `"phi"`, `"diagnostics"`, `"ppc"`, or `"pit"`.

Value

Invisibly returns `x`.

A `dcvar_covariate_summary` object (a list).

A named list with elements `mu`, `Phi`, `sigma_eps`, `beta_0`, `beta`, and, when `drift = TRUE`, `sigma_omega`.

A ggplot object.

Functions

- `print(dcvar_covariate_fit)`: Print a concise overview of the covariate DC-VAR fit.
- `summary(dcvar_covariate_fit)`: Produce a detailed summary including rho trajectory, VAR parameters, covariate effects, and diagnostics.
- `coef(dcvar_covariate_fit)`: Extract posterior means of model coefficients.
- `plot(dcvar_covariate_fit)`: Dispatch to a plot type: `"rho"`, `"phi"`, `"diagnostics"`, `"ppc"`, or `"pit"`.

`dcvar_diagnostics` *Extract MCMC diagnostics*

Description

Returns a summary of sampling diagnostics including divergences, tree depth warnings, Rhat, and effective sample size. The convergence headline is computed from sampled parameters only and excludes generated quantities and deterministic transformed outputs.

Usage

```
dcvar_diagnostics(object, ...)

## Default S3 method:
dcvar_diagnostics(object, ...)

## S3 method for class 'dcvar_model_fit'
dcvar_diagnostics(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments (unused).

Value

A named list with:

- `n_divergent`: total number of divergent transitions
- `n_max_treedepth`: transitions hitting max tree depth
- `max_rhat`: worst (highest) Rhat across sampled parameters
- `min_ess_bulk`: smallest bulk ESS among sampled parameters
- `min_ess_tail`: smallest tail ESS among sampled parameters
- `mean_accept_prob`: mean acceptance probability

`dcvar_fit-methods` *S3 methods for dcvar_fit objects*

Description

S3 methods for dcvar_fit objects

Usage

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

## S3 method for class 'dcvar_fit'
summary(object, probs = c(0.025, 0.5, 0.975), ...)

## S3 method for class 'dcvar_fit'
coef(object, ...)

## S3 method for class 'dcvar_fit'
plot(x, type = c("rho", "phi", "diagnostics", "ppc", "pit"), ...)
```

Arguments

`x, object`	A `dcvar_fit` object.
`...`	Additional arguments (unused).
`probs`	Numeric vector of quantile probabilities for the rho trajectory (default: `c(0.025, 0.5, 0.975)`).
`type`	Character; one of `"rho"`, `"phi"`, `"diagnostics"`, `"ppc"`, or `"pit"`.

Value

Invisibly returns `x`.

A `dcvar_summary` object (a list).

A named list of posterior means: `mu`, `Phi`, margin-specific scale/shape parameters (e.g. `sigma_eps` for normal margins), and `sigma_omega`.

A ggplot object.

Functions

- `print(dcvar_fit)`: Print a concise overview of the DC-VAR fit.
- `summary(dcvar_fit)`: Produce a detailed summary including rho trajectory, VAR parameters, and diagnostics.
- `coef(dcvar_fit)`: Extract posterior means of model coefficients.
- `plot(dcvar_fit)`: Dispatch to a plot type: `"rho"`, `"phi"`, `"diagnostics"`, `"ppc"`, or `"pit"`.

`dcvar_hmm` *Fit the HMM copula model*

Description

Fits a Hidden Markov Model copula VAR(1) with K discrete states and state-specific correlations. Uses ordered z_rho constraint to prevent label switching and a sticky Dirichlet prior to encourage state persistence.

Usage

```
dcvar_hmm(
  data,
  vars,
  K = 2,
  time_var = "time",
  standardize = TRUE,
  margins = "normal",
  skew_direction = NULL,
  allow_gaps = FALSE,
  prior_mu_sd = 2,
  prior_phi_sd = 0.5,
  prior_sigma_eps_rate = 1,
  prior_kappa = 10,
  prior_alpha_off = 1,
  prior_z_rho_sd = 1,
  chains = 4,
  iter_warmup = 2000,
  iter_sampling = 4000,
  adapt_delta = 0.99,
  max_treedepth = 12,
  seed = NULL,
  cores = NULL,
  refresh = 500,
  init = NULL,
  stan_file = NULL,
  backend = getOption("dcvar.backend", "auto"),
  ...
)
```

Arguments

`data`	A data frame with time series observations.
`vars`	Character vector of two variable names to model.
`K`	Number of hidden states (default: 2).
`time_var`	Name of the time column (default: `"time"`).
`standardize`	Logical; whether to z-score variables (default: `TRUE`).
`margins`	Marginal distribution specification. Either a single string applied to both variables, or a length-2 character vector giving a per-variable (mixed) margin, e.g. `c("normal", "exponential")`. Each entry is one of `"normal"` (default), `"exponential"`, `"skew_normal"`, or `"gamma"`. When the two entries differ the fit uses a generic mixed-margins Stan model under the Gaussian copula; identical entries reuse the specialised single-family model.
`skew_direction`	Integer vector of length 2 indicating skew direction for asymmetric margins. Each element must be `1` (right-skewed) or `-1` (left-skewed). Required whenever any dimension uses an `"exponential"` or `"gamma"` margin; only those dimensions consult it.
`allow_gaps`	Logical; if `FALSE` (default), interior missing values cause an error because they break VAR(1) time series adjacency. Set to `TRUE` to allow fitting with a warning instead.
`prior_mu_sd`	Prior SD for intercepts: `mu` ~ `normal(0, prior_mu_sd)`.
`prior_phi_sd`	Prior SD for VAR coefficients: `Phi` ~ `normal(0, prior_phi_sd)`.
`prior_sigma_eps_rate`	Prior mean for innovation SDs (see `prepare_dcvar_data()`).
`prior_kappa`	Sticky Dirichlet self-transition concentration (default: 10).
`prior_alpha_off`	Sticky Dirichlet off-diagonal concentration (default: 1).
`prior_z_rho_sd`	Prior SD for state-specific z_rho (default: 1.0).
`chains`	Number of MCMC chains (default: 4).
`iter_warmup`	Warmup iterations per chain (default: 2000).
`iter_sampling`	Sampling iterations per chain (default: 4000).
`adapt_delta`	Target acceptance rate (default: 0.99). The DC-VAR model uses a lower default than `dcvar_constant()` (0.999) because the non-centered parameterisation already handles posterior geometry well.
`max_treedepth`	Maximum tree depth (default: 12).
`seed`	Random seed.
`cores`	Number of parallel chains. `NULL` uses all available cores.
`refresh`	How often to print progress (default: 500). Set to 0 for silent operation.
`init`	Custom init function or `NULL` for smart defaults.
`stan_file`	Path to a custom Stan file, or `NULL` to use the bundled model.
`backend`	Character: `"auto"` (default, uses rstan), `"rstan"`, or `"cmdstanr"`. Can also be set globally via `options(dcvar.backend = "cmdstanr")`.
`...`	Additional backend-specific sampling arguments.

Value

A `dcvar_hmm_fit` object.

See Also

`dcvar()` for the smooth time-varying model, `dcvar_constant()` for the time-invariant baseline, `hmm_states()` for state extraction, `plot_hmm_states()` for visualisation, `dcvar_compare()` for LOO-CV model comparison.

Examples

```
sim <- simulate_dcvar(
  n_time = 12,
  rho_trajectory = rho_step(12),
  seed = 1
)
fit <- dcvar_hmm(
  sim$Y_df,
  vars = c("y1", "y2"),
  K = 2,
  chains = 1,
  iter_warmup = 10,
  iter_sampling = 10,
  refresh = 0,
  seed = 1
)
print(fit)
hmm_states(fit)
```

`dcvar_hmm_fit-methods` *S3 methods for dcvar_hmm_fit objects*

Description

S3 methods for dcvar_hmm_fit objects

Usage

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

## S3 method for class 'dcvar_hmm_fit'
summary(object, probs = c(0.025, 0.5, 0.975), ...)

## S3 method for class 'dcvar_hmm_fit'
coef(object, ...)
```

```
## S3 method for class 'dcvar_hmm_fit'
plot(
  x,
  type = c("rho", "states", "transition", "phi", "diagnostics", "ppc", "pit"),
  ...
)
```

Arguments

`x, object`	A `dcvar_hmm_fit` object.
`...`	Additional arguments (unused).
`probs`	Numeric vector of quantile probabilities for the rho trajectory (default: `c(0.025, 0.5, 0.975)`).
`type`	Character; one of `"rho"`, `"states"`, `"transition"`, `"phi"`, `"diagnostics"`, `"ppc"`, or `"pit"`.

Value

Invisibly returns `x`.

A `dcvar_hmm_summary` object (a list).

A named list of posterior means: `mu`, `Phi`, margin-specific scale/shape parameters (e.g. `sigma_eps` for normal margins), `z_rho`, and `rho_state`.

A ggplot object.

Functions

- `print(dcvar_hmm_fit)`: Print a concise overview of the HMM fit.
- `summary(dcvar_hmm_fit)`: Produce a detailed summary including state information, VAR parameters, and diagnostics.
- `coef(dcvar_hmm_fit)`: Extract posterior means of model coefficients including state-specific rho values.
- `plot(dcvar_hmm_fit)`: Dispatch to a plot type: `"rho"`, `"states"`, `"transition"`, `"phi"`, `"diagnostics"`, `"ppc"`, or `"pit"`.

`dcvar_multilevel` *Fit an experimental multilevel copula VAR(1) model*

Description

Fits a hierarchical copula VAR(1) model with unit-specific VAR coefficients (random effects) and a global copula correlation. Uses non-centered parameterization for the random Phi coefficients.

Usage

```
dcvar_multilevel(
  data,
  vars,
  id_var = "id",
  time_var = "time",
  center = TRUE,
  margins = "normal",
  skew_direction = NULL,
  prior_phi_bar_sd = 0.5,
  prior_tau_phi_scale = 0.2,
  prior_sigma_sd = 1,
  prior_rho_sd = 0.5,
  chains = 4,
  iter_warmup = 2000,
  iter_sampling = 4000,
  adapt_delta = 0.9,
  max_treedepth = 14,
  seed = NULL,
  cores = NULL,
  refresh = 500,
  init = NULL,
  stan_file = NULL,
  backend = getOption("dcvar.backend", "auto"),
  ...
)
```

Arguments

`data`	A data frame in long (panel) format with columns for unit ID, time, and two outcome variables.
`vars`	Character vector of two variable names to model.
`id_var`	Name of the unit/person ID column (default: `"id"`).
`time_var`	Name of the time column (default: `"time"`).
`center`	Logical; whether to person-mean center the data (default: `TRUE`). The bundled multilevel Stan model requires `center = TRUE`; set `center = FALSE` only with a custom `stan_file` that includes intercept terms.
`margins`	Marginal distribution specification. A single string applies the same family to both variables; single-family multilevel fits support `"normal"` (default) and `"exponential"` only. A length-2 character vector gives a per-variable (mixed) margin (for example `c("normal", "gamma")`); mixed fits use a generic Stan model that supports all of `"normal"`, `"exponential"`, `"skew_normal"`, and `"gamma"` per dimension, under the Gaussian copula.
`skew_direction`	Integer vector of length 2 of `1`/`-1`. Required whenever any dimension uses an `"exponential"` or `"gamma"` margin; only those dimensions consult it.
`prior_phi_bar_sd`	Prior SD for population-mean VAR coefficients.

`prior_tau_phi_scale`	Prior scale for half-t(3) on tau_phi.
`prior_sigma_sd`	Prior SD for half-normal on innovation SDs.
`prior_rho_sd`	Prior SD for normal on rho.
`chains`	Number of MCMC chains.
`iter_warmup`	Warmup iterations per chain.
`iter_sampling`	Sampling iterations per chain.
`adapt_delta`	Target acceptance rate.
`max_treedepth`	Maximum tree depth.
`seed`	Random seed.
`cores`	Number of parallel chains.
`refresh`	How often to print progress.
`init`	Custom init function or `NULL`.
`stan_file`	Custom Stan file path or `NULL`.
`backend`	Character: `"auto"` (default, uses rstan), `"rstan"`, or `"cmdstanr"`. Can also be set globally via `options(dcvar.backend = "cmdstanr")`.
`...`	Additional backend-specific sampling arguments.

Details

Experimental extension. This multilevel variant supports `fitted()` and `predict()`. PSIS-LOO is available for exponential-margin multilevel fits. PIT diagnostics are not yet implemented.

`adapt_delta` defaults to 0.90 and `max_treedepth` to 14 because the hierarchical structure with random effects benefits from deeper trees but does not require aggressive step-size adaptation.

Value

A `dcvar_multilevel_fit` object.

Note

Single-family fits support only normal and exponential margins; use a per-variable `margins` vector (for example `c("normal", "gamma")`) to access skew_normal and gamma margins via the mixed-margin model.

The bundled multilevel Stan program is defined for person-mean centered data and omits intercept terms. With the bundled model, `center = FALSE` is therefore not supported.

See Also

`random_effects()` for extracting unit-specific coefficients, `simulate_dcvar_multilevel()` for data generation.

`dcvar_multilevel_fit-methods`

S3 methods for dcvar_multilevel_fit objects

Description

S3 methods for dcvar_multilevel_fit objects

Usage

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

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

## S3 method for class 'dcvar_multilevel_fit'
coef(object, ...)

## S3 method for class 'dcvar_multilevel_fit'
plot(x, type = c("random_effects", "diagnostics"), ...)
```

Arguments

`x, object`	A `dcvar_multilevel_fit` object.
`...`	Additional arguments (unused).
`type`	Character; one of `"random_effects"`, `"diagnostics"`.

Details

Unlike single-level models (where `coef()` returns `$Phi`), the multilevel model returns hierarchical parameters:

`phi_bar` Population-mean VAR coefficients (analogous to `Phi` in single-level models, vectorised as phi11, phi12, phi21, phi22).

`tau_phi` Between-unit SD of VAR coefficients.

scale parameters `sigma` (innovation SDs) for normal margins, `sigma_exp` for exponential margins, or per-family scale/shape parameters (e.g. `sigma_eps`, `sigma_gam`, `shape_gam`) for per-variable (mixed) margins.

`rho` Copula correlation (constant across units).

Use `random_effects()` to obtain unit-specific VAR coefficients.

Value

Invisibly returns `x`.

A `dcvar_multilevel_summary` object (a list).

A named list of posterior means.

A ggplot object.

Functions

- `print(dcvar_multilevel_fit)`: Print a concise overview.
- `summary(dcvar_multilevel_fit)`: Produce a detailed summary.
- `coef(dcvar_multilevel_fit)`: Extract posterior means of population-level coefficients.
- `plot(dcvar_multilevel_fit)`: Dispatch to a plot type.

`dcvar_sem` *Fit an experimental SEM copula VAR(1) model*

Description

Fits a copula VAR(1) model with a fixed measurement model (factor loadings and measurement error SD are not estimated). Latent innovations are treated as parameters, making this model computationally intensive for large T.

Usage

```
dcvar_sem(
  data,
  indicators,
  J = NULL,
  lambda = NULL,
  sigma_e = NULL,
  margins = "normal",
  skew_direction = NULL,
  time_var = "time",
  method = c("indicator", "naive"),
  prior_mu_sd = 0.25,
  prior_phi_sd = 0.5,
  prior_sigma_sd = 0.5,
  prior_rho_sd = 0.75,
  chains = 4,
  iter_warmup = 2000,
  iter_sampling = 4000,
  adapt_delta = 0.95,
  max_treedepth = 13,
  seed = NULL,
  cores = NULL,
```

```
  refresh = 500,
  init = NULL,
  stan_file = NULL,
  backend = getOption("dcvar.backend", "auto"),
  ...
)
```

Arguments

`data`	A data frame with time series of indicator variables.
`indicators`	A list of two character vectors, each naming J indicator columns per latent variable. For example: `list(PA = c("y1_1", "y1_2", "y1_3"), NA_ = c("y2_1", "y2_2", "y2_3"))`.
`J`	Number of indicators per latent variable. If `method = "naive"` and `J` is `NULL`, it is inferred from `indicators`.
`lambda`	Numeric vector of length J with fixed factor loadings. Required when `method = "indicator"`.
`sigma_e`	Fixed measurement error SD (scalar). Required when `method = "indicator"`.
`margins`	Latent-innovation marginal specification. A single string applies the same family to both latent variables; single-family SEM fits support `"normal"` (default) and `"exponential"` only. A length-2 character vector gives a per-variable (mixed) margin (for example `c("normal", "gamma")`); mixed fits use a generic Stan model that supports all of `"normal"`, `"exponential"`, `"skew_normal"`, and `"gamma"` per dimension, under the Gaussian copula. Applies to both the indicator and naive methods.
`skew_direction`	Integer vector of length 2 of `1` (right-skewed) or `-1` (left-skewed). Required whenever any dimension uses an `"exponential"` or `"gamma"` margin; only those dimensions consult it.
`time_var`	Name of the time column (default: `"time"`).
`method`	Character string: `"indicator"` (default) fits the fixed measurement model; `"naive"` averages indicators within each latent block and fits the observed score VAR.
`prior_mu_sd`	Prior SD for intercepts: `mu ~ normal(0, prior_mu_sd)`.
`prior_phi_sd`	Prior SD for VAR coefficients: `phi ~ normal(0, prior_phi_sd)`.
`prior_sigma_sd`	Prior SD for the lognormal prior on the latent innovation scale parameter. For normal margins this is applied to `sigma`; for exponential margins it is applied to `sigma_exp`.
`prior_rho_sd`	Prior SD for rho_raw: `rho_raw ~ normal(0, prior_rho_sd)`, with `rho = 0.97 * tanh(rho_raw)`.
`chains`	Number of MCMC chains.
`iter_warmup`	Warmup iterations per chain.
`iter_sampling`	Sampling iterations per chain.
`adapt_delta`	Target acceptance rate.
`max_treedepth`	Maximum tree depth.

`seed`	Random seed.
`cores`	Number of parallel chains.
`refresh`	How often to print progress.
`init`	Custom init function or `NULL`.
`stan_file`	Custom Stan file path or `NULL`.
`backend`	Character: `"auto"` (default, uses rstan), `"rstan"`, or `"cmdstanr"`. Can also be set globally via `options(dcvar.backend = "cmdstanr")`.
`...`	Additional backend-specific sampling arguments.

Details

Experimental extension. This SEM variant supports `fitted()` and `predict()`. PSIS-LOO is available for naive score fits. PIT diagnostics are not yet implemented.

Boundary constraints. The SEM model constrains each VAR coefficient (Phi) to the interval $[-0.99, 0.99]$, unlike other dcvar models where Phi is unconstrained. Very strong autoregressive or cross-lag dynamics near ± 1 cannot be captured by this variant.

The copula correlation ρ is constrained to $(-0.97, 0.97)$ via `rho = 0.97 * tanh(rho_raw)` to avoid boundary singularity in the Gaussian copula density. Extremely high correlations near ± 1 are truncated.

Margins. Single-family SEM fits support normal and exponential latent innovation margins. Exponential margins use the same shifted-exponential parameterization as the single-level models and therefore require `skew_direction`. A per-variable (mixed) `margins` vector additionally supports `"skew_normal"` and `"gamma"` per dimension, routing to a generic mixed-margins Stan model.

Post-estimation. `fitted()` and `predict()` are available for both the latent-state scale (`type = "link"`) and the observed-indicator scale (`type = "response"`). Use `latent_states()` when you specifically need the full posterior summaries of the latent trajectories.

Value

A `dcvar_sem_fit` object.

Note

Single-family SEM fits are limited to normal and exponential latent margins. Skew-normal and gamma margins are available within `dcvar_sem` itself via a per-variable (mixed) `margins` vector (for example `c("normal", "gamma")`); only a homogeneous skew-normal or gamma latent margin requires another model family such as `dcvar()`, `dcvar_constant()`, or `dcvar_hmm()`.

See Also

`latent_states()` for extracting estimated latent states, `simulate_dcvar_sem()` for data generation.

`dcvar_sem_fit-methods` *S3 methods for dcvar_sem_fit objects*

Description

S3 methods for dcvar_sem_fit objects

Usage

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

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

## S3 method for class 'dcvar_sem_fit'
coef(object, ...)

## S3 method for class 'dcvar_sem_fit'
plot(x, type = c("latent_states", "rho", "diagnostics"), ...)
```

Arguments

`x, object`	A `dcvar_sem_fit` object.
`...`	Additional arguments (unused).
`type`	Character; one of `"latent_states"`, `"rho"`, `"diagnostics"`.

Value

Invisibly returns `x`.

A `dcvar_sem_summary` object (a list).

A named list of posterior means.

A ggplot object.

Functions

- `print(dcvar_sem_fit)`: Print a concise overview.
- `summary(dcvar_sem_fit)`: Produce a detailed summary.
- `coef(dcvar_sem_fit)`: Extract posterior means of latent VAR coefficients.
- `plot(dcvar_sem_fit)`: Dispatch to a plot type.

`dcvar_stan_path` *Get path to bundled Stan model file*

Description

Returns the file path to a Stan model file included with the package.

Usage

```
dcvar_stan_path(
  model = c("dcvar", "dcvar_covariate", "dcvar_covariate_nodrift", "hmm", "constant",
    "multilevel", "sem", "sem_naive"),
  margins = "normal",
  copula = "gaussian"
)
```

Arguments

`model`	Character string: `"dcvar"`, `"dcvar_covariate"`, `"dcvar_covariate_nodrift"`, `"hmm"`, `"constant"`, `"multilevel"`, `"sem"`, or `"sem_naive"`.
`margins`	Character string: margin type (`"normal"`, `"exponential"`, `"skew_normal"`, `"gamma"`). Default: `"normal"`.
`copula`	Character string: copula family (`"gaussian"` or `"clayton"`). Default: `"gaussian"`.

Value

File path to the Stan model file.

Examples

```
dcvar_stan_path("dcvar")
dcvar_stan_path("constant", margins = "exponential")
```

`dependence_summary` *Extract a unified dependence summary*

Description

Returns posterior summaries for Kendall's tau, using the fitted copula family to transform the model-specific dependence parameter. For Gaussian copulas, `tau = 2 / pi * asin(rho)`. For Clayton copulas, `tau = theta / (theta + 2)`.

Usage

```
dependence_summary(object, ...)

## Default S3 method:
dependence_summary(object, ...)

## S3 method for class 'dcvar_fit'
dependence_summary(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_covariate_fit'
dependence_summary(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_hmm_fit'
dependence_summary(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_constant_fit'
dependence_summary(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_multilevel_fit'
dependence_summary(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_sem_fit'
dependence_summary(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments (unused).
`probs`	Numeric vector of quantile probabilities.

Value

A data frame with columns `time`, `mean`, `sd`, and one column per requested quantile.

`draws` *Extract posterior draws*

Description

Extract posterior draws from a fitted model.

Usage

```
draws(object, ...)

## Default S3 method:
```

```
draws(object, ...)

## S3 method for class 'dcvar_model_fit'
draws(object, variable = NULL, format = "draws_array", ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments (unused).
`variable`	Character vector of parameter names. `NULL` returns all.
`format`	Draw format: `"draws_array"`, `"draws_matrix"`, or `"draws_df"` (default: `"draws_array"`).

Value

A posterior draws object.

`fitted.dcvar_model_fit`

Fitted values from a copula VAR model

Description

Returns the one-step-ahead fitted values (posterior mean of y_hat) from the VAR(1) component. For single-level fits (constant, dynamic, HMM, covariate) this is the mean-centered form `y_hat[t] = mu + Phi * (y[t-1] - mu)`.

Usage

```
## S3 method for class 'dcvar_model_fit'
fitted(object, type = c("link", "response"), ...)

## S3 method for class 'dcvar_multilevel_fit'
fitted(object, type = c("link", "response"), ...)

## S3 method for class 'dcvar_sem_fit'
fitted(object, type = c("link", "response"), ...)
```

Arguments

`object`	A fitted model object.
`type`	Character; `"link"` (default) returns values on the model's internal scale (standardized if applicable), `"response"` back-transforms to the original data scale.
`...`	Additional arguments (unused).

Details

If the model was fit with `standardize = TRUE` (the default), fitted values are on the standardized (z-scored) scale by default. Use `type = "response"` to back-transform to the original data scale.

`fitted()` and `predict()` are implemented for the public fit classes. For multilevel fits, the methods return unit-specific trajectories; because the data are already person-mean-centered, the link-scale fitted values apply the unit's VAR matrix to the previous observation with no `mu` term (`y_hat[t] = Phi_unit %*% y[t-1]`), and the person mean is re-added only for `type = "response"`. For SEM fits, `type = "link"` returns latent-state summaries and `type = "response"` returns observed indicator-scale summaries.

Value

A data frame of posterior-mean fitted values. Single-level fits return columns `time` plus one column per modeled variable. Multilevel fits additionally include `unit`. SEM fits return either latent-state columns (`type = "link"`) or observed-indicator columns (`type = "response"`).

`hmm_states` *Extract HMM state information*

Description

Returns state posteriors, Viterbi path, state-specific rho values, and the transition matrix from an HMM copula fit.

Usage

```
hmm_states(object, ...)

## Default S3 method:
hmm_states(object, ...)

## S3 method for class 'dcvar_hmm_fit'
hmm_states(object, ...)
```

Arguments

`object`	A `dcvar_hmm_fit` object.
`...`	Additional arguments (unused).

Value

A named list with:

- `gamma`: T_eff x K matrix of posterior state probabilities
- `viterbi`: integer vector of MAP state sequence
- `rho_state`: list with `mean`, `lower`, `upper` for each state

- `A`: K x K posterior mean transition matrix
- `rho_hmm`: posterior-averaged rho trajectory

`interpret_rho_trajectory` *Interpret a rho trajectory in clinical terms*

Description

Generates a human-readable interpretation of the estimated rho trajectory, describing the overall trend, magnitude of change, and key features.

Usage

```
interpret_rho_trajectory(
  object,
  threshold = 0.1,
  strength_breaks = .default_strength_breaks,
  magnitude_breaks = .default_magnitude_breaks,
  fluctuation_threshold = 0.3,
  ...
)
```

Arguments

`object`	A fitted model object (`dcvar_fit`, `dcvar_covariate_fit`, `dcvar_hmm_fit`, `dcvar_constant_fit`, `dcvar_multilevel_fit`, or `dcvar_sem_fit`).
`threshold`	Minimum absolute change in posterior-mean rho to be considered "meaningful" (default: 0.1).
`strength_breaks`	Named numeric vector of thresholds for classifying correlation strength (default: `c(strong = 0.7, moderate = 0.4, weak = 0.2)`). Values above the highest threshold are "strong", etc.; values at or below the lowest threshold are classified as "negligible".
`magnitude_breaks`	Named numeric vector of thresholds for classifying the magnitude of trajectory range (default: `c(large = 0.5, moderate = 0.3, small = 0.1)`). Values at or below the lowest threshold are classified as "negligible".
`fluctuation_threshold`	Proportion of sign changes in first differences to flag "substantial fluctuation" (default: 0.3).
`...`	Additional arguments (unused).

Value

A character string with the interpretation (invisibly). The interpretation is also printed to the console.

Examples

```
sim <- simulate_dcvar(
  n_time = 12,
  rho_trajectory = rho_decreasing(12),
  seed = 1
)
fit <- dcvar(
  sim$Y_df,
  vars = c("y1", "y2"),
  chains = 1,
  iter_warmup = 10,
  iter_sampling = 10,
  refresh = 0,
  seed = 1
)
interpret_rho_trajectory(fit)
```

`latent_states` *Extract latent states from a SEM fit*

Description

Returns posterior summaries for the estimated latent states at each time point.

Usage

```
latent_states(object, ...)

## Default S3 method:
latent_states(object, ...)

## S3 method for class 'dcvar_sem_fit'
latent_states(object, probs = c(0.025, 0.5, 0.975), ...)
```

Arguments

`object`	A `dcvar_sem_fit` object.
`...`	Additional arguments (unused).
`probs`	Numeric vector of quantile probabilities.

Value

A data frame with columns `time`, `variable`, `mean`, `sd`, and quantile columns.

`loo.dcvar` *Compute LOO-CV for a fitted model*

Description

Compute LOO-CV for a fitted model

Usage

```
## S3 method for class 'dcvar_fit'
loo(x, ...)

## S3 method for class 'dcvar_covariate_fit'
loo(x, ...)

## S3 method for class 'dcvar_hmm_fit'
loo(x, ...)

## S3 method for class 'dcvar_constant_fit'
loo(x, ...)

## S3 method for class 'dcvar_multilevel_fit'
loo(x, ...)

## S3 method for class 'dcvar_sem_fit'
loo(x, ...)
```

Arguments

`x`	A fitted model object.
`...`	Additional arguments passed to `loo::loo()`.

Details

PSIS-LOO is available for Gaussian and Clayton single-level fits, covariate fits, exponential-margin multilevel fits, and naive SEM score fits. Indicator SEM fits and normal-margin multilevel fits are not supported because their stored `log_lik` quantities are not comparable pointwise predictive densities.

Value

A `loo` object from the loo package.

`pit_test` *KS test for PIT uniformity*

Description

Runs a Kolmogorov-Smirnov test per variable to assess whether PIT values are approximately uniform. This is a heuristic check on the plug-in PIT values returned by `pit_values()`, not an exact posterior predictive test.

Usage

```
pit_test(object, ...)

## Default S3 method:
pit_test(object, ...)

## S3 method for class 'dcvar_model_fit'
pit_test(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments (unused).

Details

This applies a Kolmogorov-Smirnov test to the approximate PIT values returned by `pit_values()`. The result is a heuristic check and does not account for serial dependence or full posterior uncertainty. PIT tests are currently implemented for single-level fits only.

Value

A data frame with columns `variable`, `ks_statistic`, `p_value`, `n`.

`pit_values` *Extract PIT values from a fitted model*

Description

Computes approximate Probability Integral Transform values using posterior mean residuals and posterior mean margin parameters. Large departures from uniformity can indicate model misfit, but these are not exact posterior predictive PIT values.

Usage

```
pit_values(object, ...)

## Default S3 method:
pit_values(object, ...)

## S3 method for class 'dcvar_model_fit'
pit_values(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments (unused).

Details

PIT values are computed from posterior mean residuals and posterior mean margin parameters. Treat them as a fast plug-in diagnostic rather than an exact posterior predictive transform that integrates over full posterior uncertainty. PIT diagnostics are currently implemented for the three core single-level fit classes only.

Value

A data frame with columns `time`, `variable`, `pit`.

`plot_diagnostics` *Plot MCMC diagnostics*

Description

Creates a combined panel with trace plots, Rhat, and ESS diagnostics.

Usage

```
plot_diagnostics(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments (unused).

Value

A combined ggplot object (via patchwork).

`plot_hmm_states` *Plot HMM state posteriors*

Description

Plot HMM state posteriors

Usage

```
plot_hmm_states(object, show_viterbi = TRUE, ...)
```

Arguments

`object`	A `dcvar_hmm_fit` object.
`show_viterbi`	Logical; overlay the Viterbi (MAP) state sequence (default: `TRUE`).
`...`	Additional arguments (unused).

Value

A ggplot object.

`plot_latent_states` *Plot estimated latent states with credible intervals*

Description

Plot estimated latent states with credible intervals

Usage

```
plot_latent_states(object, true_states = NULL, ...)
```

Arguments

`object`	A `dcvar_sem_fit` object.
`true_states`	Optional T x 2 matrix of true latent states for overlay.
`...`	Additional arguments (unused).

Value

A ggplot object.

`plot_phi` *Plot VAR(1) coefficient matrix as a heatmap*

Description

Plot VAR(1) coefficient matrix as a heatmap

Usage

```
plot_phi(object, var_names = NULL, ...)
```

Arguments

`object`	A fitted model object.
`var_names`	Character vector of variable names for axis labels.
`...`	Additional arguments (unused).

Value

A ggplot object.

`plot_pit` *Plot PIT histograms*

Description

Creates faceted histograms of the approximate PIT values returned by `pit_values()`. Under good model fit, these histograms should be roughly uniform, but they remain plug-in diagnostics rather than exact posterior predictive checks.

Usage

```
plot_pit(object, bins = 20, ...)
```

Arguments

`object`	A fitted model object.
`bins`	Number of histogram bins (default: 20).
`...`	Additional arguments (unused).

Details

PIT histograms visualize the approximate plug-in PIT values returned by `pit_values()`. They are currently implemented for the three core single-level fit classes only.

Value

A ggplot object.

`plot_ppc` *Posterior predictive check for residual correlations*

Description

Posterior predictive check for residual correlations

Usage

```
plot_ppc(object, n_sample = 100, ...)
```

Arguments

`object`	A fitted model object.
`n_sample`	Number of posterior draws to use (default: 100).
`...`	Additional arguments (unused).

Details

Posterior predictive checks are currently available for normal and exponential margins. Gamma and skew-normal fits store copula-level replicated z-scores in `eps_rep`, so their replicated draws are not on the same residual scale as `eps`.

Value

A ggplot object.

`plot_random_effects` *Plot random effects (caterpillar plot)*

Description

Displays unit-specific VAR coefficients with credible intervals.

Usage

```
plot_random_effects(object, ...)
```

Arguments

`object`	A `dcvar_multilevel_fit` object.
`...`	Additional arguments (unused).

Value

A ggplot object.

`plot_rho` *Plot the rho trajectory with credible intervals*

Description

Plot the rho trajectory with credible intervals

Usage

```
plot_rho(
  object,
  show_ci = TRUE,
  ci_level = 0.95,
  inner_level = 0.8,
  true_rho = NULL,
  title = NULL,
  ...
)
```

Arguments

`object`	A fitted model object with a rho trajectory (`dcvar_fit`, `dcvar_covariate_fit`, `dcvar_hmm_fit`, or `dcvar_constant_fit`).
`show_ci`	Logical; show credible interval ribbons (default: `TRUE`).
`ci_level`	Credible interval level for the outer ribbon (default: 0.95).
`inner_level`	Credible interval level for the inner ribbon (default: 0.80). Set to `NULL` to disable the inner ribbon.
`true_rho`	Optional numeric vector of true rho values for overlay (useful for simulation studies).
`title`	Plot title.
`...`	Additional arguments (unused).

Value

A ggplot object.

`plot_trajectories` *Plot and compare multiple rho trajectory shapes*

Description

Visualises several named trajectory scenarios side by side for comparison.

Usage

```
plot_trajectories(
  n_time,
 scenarios = c("constant", "decreasing", "increasing", "random_walk", "single_middle",
    "large_change", "double_relapse"),
  ...
)
```

Arguments

`n_time`	Number of time points.
`scenarios`	Character vector of scenario names (see `rho_scenario()`). Default: all built-in scenarios.
`...`	Additional arguments passed to `rho_scenario()`.

Value

A ggplot object.

Examples

```
plot_trajectories(100)
plot_trajectories(100, scenarios = c("decreasing", "single_middle"))
```

`predict.dcvar_model_fit` *One-step-ahead predictions from a copula VAR model*

Description

Returns point predictions and **marginal** prediction intervals by combining the VAR(1) fitted values with the estimated innovation SDs. Intervals are computed per-variable using a normal approximation and do not account for the copula dependence structure between variables.

Usage

```
## S3 method for class 'dcvar_model_fit'
predict(object, type = c("link", "response"), ci_level = 0.95, ...)

## S3 method for class 'dcvar_multilevel_fit'
predict(object, type = c("link", "response"), ci_level = 0.95, ...)

## S3 method for class 'dcvar_sem_fit'
predict(object, type = c("link", "response"), ci_level = 0.95, ...)
```

Arguments

`object`	A fitted model object.
`type`	Character; `"link"` (default) returns values on the model's internal scale (standardized if applicable), `"response"` back-transforms to the original data scale.
`ci_level`	Prediction interval level (default: 0.95).
`...`	Additional arguments (unused).

Details

`predict()` is implemented for the public fit classes. For multilevel fits, the methods return unit-specific trajectories. For SEM fits, `type = "link"` returns latent states and `type = "response"` returns observed indicator predictions.

Value

A data frame of marginal prediction intervals at the specified level. Single-level and SEM fits return columns `time`, `variable`, `mean`, `lower`, `upper`. Multilevel fits additionally include `unit`.

`prepare_constant_data` *Prepare data for the constant copula model*

Description

Transforms a data frame into a list suitable for the constant copula Stan model.

Usage

```
prepare_constant_data(
  data,
  vars,
  time_var = "time",
  standardize = TRUE,
  margins = "normal",
  skew_direction = NULL,
  prior_mu_sd = 2,
```

```
    prior_phi_sd = 0.5,
    prior_sigma_eps_rate = 1,
    prior_z_rho_sd = 1,
    allow_gaps = FALSE
  )
```

Arguments

`data`	A data frame with time series observations.
`vars`	Character vector of two variable names to model.
`time_var`	Name of the time column (default: `"time"`).
`standardize`	Logical; whether to z-score variables (default: `TRUE`).
`margins`	Marginal distribution specification. Either a single string applied to both variables, or a length-2 character vector giving a per-variable (mixed) margin (for example `c("normal", "exponential")`). Each entry is one of `"normal"` (default), `"exponential"`, `"skew_normal"`, or `"gamma"`. When the two entries differ the returned data carries an integer `family` array for the generic mixed-margins Stan model; identical entries are equivalent to the scalar form and reuse the specialised single-family model.
`skew_direction`	Integer vector of length 2 of `1` (right-skewed) or `-1` (left-skewed). Required whenever any dimension uses an `"exponential"` or `"gamma"` margin; only those dimensions consult it.
`prior_mu_sd`	Prior SD for intercepts: `mu` ~ `normal(0, prior_mu_sd)`.
`prior_phi_sd`	Prior SD for VAR coefficients: `Phi` ~ `normal(0, prior_phi_sd)`.
`prior_sigma_eps_rate`	Prior mean for innovation SDs: `sigma_eps` ~ `exponential(1/prior_sigma_eps_rate)`. Default `1` gives `exponential(1)` with prior mean 1.
`prior_z_rho_sd`	Prior SD for rho on Fisher-z scale (default: 1.0).
`allow_gaps`	Logical; if `FALSE` (default), interior missing values cause an error. If `TRUE`, they produce a warning and are removed.

Value

A named list suitable as Stan data input.

`prepare_dcvar_covariate_data`

Prepare data for the covariate DC-VAR model

Description

Transforms a data frame into a list suitable for the Gaussian covariate DC-VAR Stan models. Outcome rows are sorted by `time_var`; rows with missing outcomes are removed with the same adjacency rules used by `prepare_dcvar_data()`, and covariates are filtered in the same order so `X[t + 1, ]` aligns with the outcome occasion of transition `Y[t, ]` -> `Y[t + 1, ]`.

Usage

```
prepare_dcvar_covariate_data(
  data,
  vars,
  covariates,
  time_var = "time",
  standardize = TRUE,
  standardize_covariates = FALSE,
  allow_gaps = FALSE,
  prior_mu_sd = 2,
  prior_phi_sd = 0.5,
  prior_sigma_eps_rate = 1,
  prior_sigma_omega_rate = 0.1,
  prior_rho_init_sd = 1,
  prior_beta_sd = 1,
  zero_init_eta = TRUE
)
```

Arguments

`data`	A data frame with time series observations.
`vars`	Character vector of two variable names to model.
`covariates`	Character vector of covariate column names.
`time_var`	Name of the time column (default: `"time"`).
`standardize`	Logical; whether to z-score variables (default: `TRUE`).
`standardize_covariates`	Logical; whether to z-score covariates (default: `FALSE`). Binary phase indicators should usually be left on their original scale.
`allow_gaps`	Logical; if `FALSE` (default), interior missing values cause an error. If `TRUE`, they produce a warning and are removed.
`prior_mu_sd`	Prior SD for intercepts: `mu` ~ `normal(0, prior_mu_sd)`.
`prior_phi_sd`	Prior SD for VAR coefficients: `Phi` ~ `normal(0, prior_phi_sd)`.
`prior_sigma_eps_rate`	Prior mean for innovation SDs: `sigma_eps` ~ `exponential(1/prior_sigma_eps_rate)`. Default `1` gives `exponential(1)` with prior mean 1.
`prior_sigma_omega_rate`	Prior mean for rho process SD: `sigma_omega` ~ `exponential(1/prior_sigma_omega_rate)`. Default `0.1` gives `exponential(10)` with prior mean 0.1.
`prior_rho_init_sd`	Prior SD for initial rho on Fisher-z scale.
`prior_beta_sd`	Prior SD for the covariate effects: `beta` ~ `normal(0, prior_beta_sd)`.
`zero_init_eta`	Logical; if `TRUE` (default), fixes the first residual drift state at zero (`eta[1] = 0`). If `FALSE`, the first transition can receive an immediate residual random-walk shock.

Value

A named list suitable as Stan data input.

`prepare_dcvar_data` *Prepare data for the DC-VAR model*

Description

Transforms a data frame into a list suitable for the DC-VAR Stan model. Handles sorting, missing values, and optional standardization.

Usage

```
prepare_dcvar_data(
  data,
  vars,
  time_var = "time",
  standardize = TRUE,
  margins = "normal",
  skew_direction = NULL,
  prior_mu_sd = 2,
  prior_phi_sd = 0.5,
  prior_sigma_eps_rate = 1,
  prior_sigma_omega_rate = 0.1,
  prior_rho_init_sd = 1,
  allow_gaps = FALSE
)
```

Arguments

`data`	A data frame with time series observations.
`vars`	Character vector of two variable names to model.
`time_var`	Name of the time column (default: `"time"`).
`standardize`	Logical; whether to z-score variables (default: `TRUE`).
`margins`	Marginal distribution specification. Either a single string applied to both variables, or a length-2 character vector giving a per-variable (mixed) margin (for example `c("normal", "exponential")`). Each entry is one of `"normal"` (default), `"exponential"`, `"skew_normal"`, or `"gamma"`. When the two entries differ the returned data carries an integer `family` array for the generic mixed-margins Stan model; identical entries are equivalent to the scalar form and reuse the specialised single-family model.
`skew_direction`	Integer vector of length D indicating skew direction for asymmetric margins. Each element must be `1` (right-skewed) or `-1` (left-skewed). Required whenever any dimension uses an `"exponential"` or `"gamma"` margin.
`prior_mu_sd`	Prior SD for intercepts: `mu` ~ `normal(0, prior_mu_sd)`.

`prior_phi_sd`	Prior SD for VAR coefficients: `Phi ~ normal(0, prior_phi_sd)`.
`prior_sigma_eps_rate`	Prior mean for innovation SDs: `sigma_eps ~ exponential(1/prior_sigma_eps_rate)`. Default `1` gives `exponential(1)` with prior mean 1.
`prior_sigma_omega_rate`	Prior mean for rho process SD: `sigma_omega ~ exponential(1/prior_sigma_omega_rate)`. Default `0.1` gives `exponential(10)` with prior mean 0.1.
`prior_rho_init_sd`	Prior SD for initial rho on Fisher-z scale.
`allow_gaps`	Logical; if `FALSE` (default), interior missing values cause an error. If `TRUE`, they produce a warning and are removed.

Value

A named list suitable as Stan data input.

Prior naming conventions

Parameters ending in `_sd` specify normal prior standard deviations (location parameters). Parameters ending in `_rate` specify exponential prior means (scale parameters), where the exponential rate is `1/prior_*_rate`. The constant and HMM models use `prior_z_rho_sd` (normal prior on the Fisher-z scale), while the DC-VAR model uses `prior_sigma_omega_rate` (exponential prior on the random-walk SD) because the two quantities have fundamentally different roles.

`prepare_hmm_data` *Prepare data for the HMM copula model*

Description

Transforms a data frame into a list suitable for the HMM copula Stan model. Includes HMM-specific prior hyperparameters.

Usage

```
prepare_hmm_data(
  data,
  vars,
  K = 2,
  time_var = "time",
  standardize = TRUE,
  margins = "normal",
  skew_direction = NULL,
  prior_mu_sd = 2,
  prior_phi_sd = 0.5,
  prior_sigma_eps_rate = 1,
  prior_kappa = 10,
```

```
  prior_alpha_off = 1,
  prior_z_rho_sd = 1,
  allow_gaps = FALSE
)
```

Arguments

`data`	A data frame with time series observations.
`vars`	Character vector of two variable names to model.
`K`	Number of hidden states (default: 2).
`time_var`	Name of the time column (default: `"time"`).
`standardize`	Logical; whether to z-score variables (default: `TRUE`).
`margins`	Marginal distribution specification. Either a single string applied to both variables, or a length-2 character vector giving a per-variable (mixed) margin (for example `c("normal", "exponential")`). Each entry is one of `"normal"` (default), `"exponential"`, `"skew_normal"`, or `"gamma"`. When the two entries differ the returned data carries an integer `family` array for the generic mixed-margins Stan model; identical entries are equivalent to the scalar form and reuse the specialised single-family model.
`skew_direction`	Integer vector of length D indicating skew direction for asymmetric margins. Each element must be `1` (right-skewed) or `-1` (left-skewed). Required whenever any dimension uses an `"exponential"` or `"gamma"` margin.
`prior_mu_sd`	Prior SD for intercepts: `mu ~ normal(0, prior_mu_sd)`.
`prior_phi_sd`	Prior SD for VAR coefficients: `Phi ~ normal(0, prior_phi_sd)`.
`prior_sigma_eps_rate`	Prior mean for innovation SDs: `sigma_eps ~ exponential(1/prior_sigma_eps_rate)`. Default `1` gives `exponential(1)` with prior mean 1.
`prior_kappa`	Sticky Dirichlet self-transition concentration (default: 10).
`prior_alpha_off`	Sticky Dirichlet off-diagonal concentration (default: 1).
`prior_z_rho_sd`	Prior SD for state-specific z_rho values (default: 1.0).
`allow_gaps`	Logical; if `FALSE` (default), interior missing values cause an error. If `TRUE`, they produce a warning and are removed.

Value

A named list suitable as Stan data input.

---

prepare_multilevel_data

Prepare data for the multilevel copula VAR model

Description

Prepare data for the multilevel copula VAR model

Usage

```
prepare_multilevel_data(
  data,
  vars,
  id_var = "id",
  time_var = "time",
  center = TRUE,
  prior_phi_bar_sd = 0.5,
  prior_tau_phi_scale = 0.2,
  prior_sigma_sd = 1,
  prior_rho_sd = 0.5,
  margins = "normal",
  skew_direction = NULL
)
```

Arguments

`data`	A data frame in long (panel) format.
`vars`	Character vector of two variable names.
`id_var`	Name of the unit/person ID column.
`time_var`	Name of the time column.
`center`	Logical scalar; person-mean center (default: `TRUE`).
`prior_phi_bar_sd`	Prior SD for phi_bar.
`prior_tau_phi_scale`	Prior scale for tau_phi.
`prior_sigma_sd`	Prior SD for sigma.
`prior_rho_sd`	Prior SD for rho.
`margins`	Marginal distribution specification. A single string is restricted to `"normal"` (default) or `"exponential"`. A length-2 character vector gives a per-variable (mixed) margin that may combine any of `"normal"`, `"exponential"`, `"skew_normal"`, and `"gamma"` per dimension, routing to the generic `multilevel_mixed` Stan model.
`skew_direction`	Length-2 integer vector of +1/-1. Required whenever any dimension uses an `"exponential"` or `"gamma"` margin.

Value

A named list suitable as Stan data input.

`prepare_sem_data` *Prepare data for the SEM copula VAR model*

Description

Transforms a data frame of indicator variables into a list suitable for the SEM copula Stan model. The measurement model parameters (lambda, sigma_e) are fixed and passed through to Stan.

Usage

```
prepare_sem_data(
  data,
  indicators,
  J = NULL,
  lambda = NULL,
  sigma_e = NULL,
  margins = "normal",
  skew_direction = NULL,
  time_var = "time",
  prior_mu_sd = 0.25,
  prior_phi_sd = 0.5,
  prior_sigma_sd = 0.5,
  prior_rho_sd = 0.75,
  method = c("indicator", "naive")
)
```

Arguments

`data`	A data frame with time series of indicator variables.
`indicators`	A list of two character vectors, each naming J indicator columns per latent variable.
`J`	Number of indicators per latent variable.
`lambda`	Numeric vector of length J with fixed factor loadings.
`sigma_e`	Fixed measurement error SD (scalar).
`margins`	Latent innovation margin specification. A single string is restricted to `"normal"` (default) or `"exponential"`. A length-2 character vector gives a per-variable (mixed) margin that may combine any of `"normal"`, `"exponential"`, `"skew_normal"`, and `"gamma"` per dimension, routing to the generic `sem_mixed` / `sem_naive_mixed` Stan model.
`skew_direction`	Integer vector of length 2 of +1/-1. Required whenever any dimension uses an `"exponential"` or `"gamma"` margin.

`time_var`	Name of the time column (default: `"time"`).
`prior_mu_sd`	Prior SD for intercepts.
`prior_phi_sd`	Prior SD for VAR coefficients.
`prior_sigma_sd`	Prior SD for the lognormal prior on the latent innovation scale parameter.
`prior_rho_sd`	Prior SD for rho_raw.
`method`	Character string: `"indicator"` for the fixed measurement model or `"naive"` for row-mean factor scores.

Value

A named list suitable as Stan data input.

`print.dcvar_constant_summary`

Print a dcvar_constant_summary object

Description

Print a dcvar_constant_summary object

Usage

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

Arguments

`x`	A `dcvar_constant_summary` object as returned by `summary.dcvar_constant_fit()`.
`...`	Additional arguments (unused).

Value

Invisibly returns `x`.

`print.dcvar_covariate_summary`

Print a dcvar_covariate_summary object

Description

Print a dcvar_covariate_summary object

Usage

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

Arguments

`x`	A `dcvar_covariate_summary` object as returned by `summary()`.
`...`	Additional arguments (unused).

Value

Invisibly returns `x`.

`print.dcvar_hmm_summary`

Print a dcvar_hmm_summary object

Description

Print a dcvar_hmm_summary object

Usage

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

Arguments

`x`	A `dcvar_hmm_summary` object as returned by `summary.dcvar_hmm_fit()`.
`...`	Additional arguments (unused).

Value

Invisibly returns `x`.

`print.dcvar_multilevel_summary`

Print a dcvar_multilevel_summary object

Description

Print a dcvar_multilevel_summary object

Usage

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

Arguments

`x`	A `dcvar_multilevel_summary` object.
`...`	Additional arguments (unused).

Value

Invisibly returns `x`.

`print.dcvar_sem_summary`

Print a dcvar_sem_summary object

Description

Print a dcvar_sem_summary object

Usage

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

Arguments

`x`	A `dcvar_sem_summary` object.
`...`	Additional arguments (unused).

Value

Invisibly returns `x`.

`print.dcvar_summary` *Print a dcvar_summary object*

Description

Print a dcvar_summary object

Usage

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

Arguments

`x`	A `dcvar_summary` object as returned by `summary.dcvar_fit()`.
`...`	Additional arguments (unused).

Value

Invisibly returns `x`.

`random_effects` *Extract random effects from a multilevel fit*

Description

Returns posterior summaries for unit-specific VAR coefficients.

Usage

```
random_effects(object, ...)

## Default S3 method:
random_effects(object, ...)

## S3 method for class 'dcvar_multilevel_fit'
random_effects(object, ...)
```

Arguments

`object`	A `dcvar_multilevel_fit` object.
`...`	Additional arguments (unused).

Value

A data frame with columns `unit`, `parameter`, `mean`, `sd`, `q2.5`, `q97.5`.

`rho_constant` *Generate a constant rho trajectory*

Description

Generate a constant rho trajectory

Usage

```
rho_constant(n_time, rho = 0.5)
```

Arguments

`n_time`	Number of time points.
`rho`	Constant correlation value (default: 0.5). Must be in [-1, 1].

Value

Numeric vector of length `n_time - 1`.

Examples

```
rho_constant(100, rho = 0.5)
```

`rho_decreasing` *Generate a logistically decreasing rho trajectory*

Description

Mimics a therapy effect where coupling decreases from high to low.

Usage

```
rho_decreasing(
  n_time,
  rho_start = 0.7,
  rho_end = 0.3,
  midpoint = NULL,
  steepness = 0.05
)
```

Arguments

`n_time`	Number of time points.
`rho_start`	Starting rho value (default: 0.7). Must be in [-1, 1].
`rho_end`	Ending rho value (default: 0.3). Must be in [-1, 1].
`midpoint`	Time point of inflection (default: `n_time/2`).
`steepness`	Controls transition sharpness (default: 0.05).

Value

Numeric vector of length `n_time - 1`.

Examples

```
rho_decreasing(100)
```

`rho_double_step` *Generate a double-breakpoint (relapse pattern) rho trajectory*

Description

Three-phase trajectory: level A -> level B -> level C.

Usage

```
rho_double_step(
  n_time,
  rho_levels = c(0.7, 0.3, 0.7),
  breakpoints = c(1/3, 2/3),
  transition_width = 0
)
```

Arguments

`n_time`	Number of time points.
`rho_levels`	Numeric vector of three rho levels (default: `c(0.7, 0.3, 0.7)`).
`breakpoints`	Numeric vector of two breakpoint positions (default: `c(1/3, 2/3)`). Interpreted as proportions of `n_time - 1` if <= 1.
`transition_width`	Number of time points for smooth transitions (default: 0).

Value

Numeric vector of length `n_time - 1`.

Examples

```
rho_double_step(100, rho_levels = c(0.7, 0.3, 0.7))
```

`rho_increasing` *Generate a logistically increasing rho trajectory*

Description

Mimics deterioration where coupling increases from low to high.

Usage

```
rho_increasing(
  n_time,
  rho_start = 0.3,
  rho_end = 0.7,
  midpoint = NULL,
  steepness = 0.05
)
```

Arguments

`n_time`	Number of time points.
`rho_start`	Starting rho value (default: 0.3). Must be in [-1, 1].
`rho_end`	Ending rho value (default: 0.7). Must be in [-1, 1].
`midpoint`	Time point of inflection (default: `n_time/2`).
`steepness`	Controls transition sharpness (default: 0.05).

Value

Numeric vector of length `n_time - 1`.

Examples

```
rho_increasing(100)
```

`rho_random_walk` *Generate a random walk rho trajectory on the Fisher-z scale*

Description

Stochastic trajectory matching the DC-VAR data-generating process.

Usage

```
rho_random_walk(n_time, z_init = 0.5, sigma_omega = 0.05, seed = NULL)
```

Arguments

`n_time`	Number of time points.
`z_init`	Initial value on Fisher-z scale (default: 0.5, corresponding to rho = 0.46).
`sigma_omega`	Innovation SD for the random walk (default: 0.05).
`seed`	Random seed for reproducibility.

Value

Numeric vector of length `n_time - 1`.

Examples

```
rho_random_walk(100, seed = 42)
```

`rho_scenario` *Get a named trajectory scenario*

Description

Convenience function to retrieve a standard scenario by name.

Usage

```
rho_scenario(scenario, n_time, ...)
```

Arguments

`scenario`	Character string. One of: • Smooth: `"constant"`, `"decreasing"`, `"increasing"`, `"random_walk"` • Step: `"single_middle"`, `"large_change"`, `"small_change"`, `"increase"`, `"double_relapse"`
`n_time`	Number of time points.
`...`	Additional arguments passed to the generator.

Value

Numeric vector of length `n_time - 1`.

Examples

```
rho_scenario("decreasing", n_time = 100)
rho_scenario("double_relapse", n_time = 150)
```

`rho_step` *Generate a single-breakpoint (step function) rho trajectory*

Description

Abrupt change from one rho level to another at a specified time.

Usage

```
rho_step(
  n_time,
  rho_before = 0.7,
  rho_after = 0.3,
  breakpoint = 0.5,
  transition_width = 0
)
```

Arguments

`n_time`	Number of time points.
`rho_before`	Rho before breakpoint (default: 0.7).
`rho_after`	Rho after breakpoint (default: 0.3).
`breakpoint`	Breakpoint location as a proportion of `n_time - 1` (if <= 1) or an absolute time index (default: 0.5).
`transition_width`	Number of time points for smooth transition. 0 = abrupt (default: 0).

Value

Numeric vector of length `n_time - 1`.

Examples

```
rho_step(100, rho_before = 0.7, rho_after = 0.3)
```

`rho_trajectory` *Extract the rho trajectory with credible intervals*

Description

Returns a data frame with the posterior mean, SD, and quantiles of the time-varying correlation at each time point.

Usage

```
rho_trajectory(object, ...)

## Default S3 method:
rho_trajectory(object, ...)

## S3 method for class 'dcvar_fit'
rho_trajectory(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_covariate_fit'
rho_trajectory(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_hmm_fit'
rho_trajectory(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_constant_fit'
rho_trajectory(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_multilevel_fit'
rho_trajectory(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)

## S3 method for class 'dcvar_sem_fit'
rho_trajectory(object, probs = c(0.025, 0.1, 0.5, 0.9, 0.975), ...)
```

Arguments

`object`	A fitted model object (`dcvar_fit`, `dcvar_covariate_fit`, `dcvar_hmm_fit`, `dcvar_constant_fit`, `dcvar_multilevel_fit`, or `dcvar_sem_fit`).
`...`	Additional arguments (unused).
`probs`	Numeric vector of quantile probabilities (default: `c(0.025, 0.1, 0.5, 0.9, 0.975)`).

Value

A data frame with columns `time`, `mean`, `sd`, and one column per quantile (e.g., `q2.5`, `q10`, `q50`, `q90`, `q97.5`). For `dcvar_constant_fit` objects, the constant rho is expanded to all `n_time - 1` time points for consistency with the time-varying models.

See Also

`plot_rho()` to visualise the trajectory, `interpret_rho_trajectory()` for a text-based summary, `var_params()` for VAR parameter extraction.

---

simulate_breakpoint_data

Simulate data with a breakpoint rho trajectory

Description

Convenience wrapper that combines `rho_step()` or `rho_double_step()` with `simulate_dcvar()` for quick breakpoint simulation studies.

Usage

```
simulate_breakpoint_data(
  n_time,
  type = c("single", "double"),
  rho_before = 0.7,
  rho_after = 0.3,
  rho_levels = c(0.7, 0.3, 0.7),
  breakpoint = 0.5,
  breakpoints = c(1/3, 2/3),
  transition_width = 0,
  mu = c(0, 0),
  Phi = matrix(c(0.3, 0.1, 0.1, 0.3), 2, 2),
  sigma_eps = c(1, 1),
  seed = NULL
)
```

Arguments

`n_time`	Number of time points.
`type`	Character; one of `"single"` (single breakpoint) or `"double"` (double breakpoint / relapse pattern).
`rho_before`	Rho before breakpoint (default: 0.7).
`rho_after`	Rho after breakpoint (default: 0.3).
`rho_levels`	Numeric vector of three rho levels for double breakpoint (default: `c(0.7, 0.3, 0.7)`). Only used when `type = "double"`.
`breakpoint`	Breakpoint location as proportion of `n_time - 1` (default: 0.5).
`breakpoints`	Numeric vector of two breakpoints for double type (default: `c(1/3, 2/3)`).
`transition_width`	Number of time points for smooth transition (default: 0 = abrupt).
`mu`	Intercept vector of length 2 (default: `c(0, 0)`).
`Phi`	VAR(1) coefficient matrix, 2x2 (default: `matrix(c(0.3, 0.1, 0.1, 0.3), 2, 2)`).
`sigma_eps`	Innovation SDs, length 2 (default: `c(1, 1)`).
`seed`	Random seed.

Value

A named list as returned by `simulate_dcvar()`. Its `true_params` additionally records the breakpoint specification: `type`, plus `breakpoint` (single) or `breakpoints` (double), as proportions of `n_time - 1`.

Examples

```
sim <- simulate_breakpoint_data(n_time = 100, type = "single", seed = 42)
plot(sim$true_params$rho, type = "l")
```

`simulate_dcvar` *Simulate data from a copula VAR(1) model*

Description

Generates bivariate time series data with correlated innovations driven by a specified rho trajectory.

Usage

```
simulate_dcvar(
  n_time,
  rho_trajectory,
  mu = c(0, 0),
  Phi = matrix(c(0.3, 0.1, 0.1, 0.3), 2, 2),
  sigma_eps = c(1, 1),
  margins = "normal",
  skew_direction = NULL,
  skew_params = NULL,
  seed = NULL
)
```

Arguments

`n_time`	Number of time points.
`rho_trajectory`	Numeric vector of length `n_time - 1` specifying the correlation at each time step. Use `rho_constant()`, `rho_decreasing()`, etc.
`mu`	Intercept vector of length 2 (default: `c(0, 0)`).
`Phi`	VAR(1) coefficient matrix, 2x2 (default: `matrix(c(0.3, 0.1, 0.1, 0.3), 2, 2)`).
`sigma_eps`	Innovation standard deviations, length 2 (default: `c(1, 1)`). Used for normal margins.
`margins`	Marginal family. Either a single string applied to both variables, or a length-2 character vector for per-variable (mixed) margins, e.g. `c("normal", "exponential")`. Each entry is one of `"normal"` (default), `"exponential"`, `"skew_normal"`, or `"gamma"`.

`skew_direction`	Length-2 integer vector of +1/-1. Required whenever any dimension uses an `"exponential"` or `"gamma"` margin; only those dimensions consult it.
`skew_params`	Named list of margin-specific parameters. `alpha` (length-2 vector of skew-normal shape params) is used by skew-normal dimensions; `shape` (scalar gamma shape parameter) is used by gamma dimensions. Both may be supplied together for mixed margins.
`seed`	Random seed for reproducibility.

Value

A named list with:

- `Y`: `n_time x 2` observation matrix
- `Y_df`: data frame with columns `time`, `y1`, `y2` (ready for `dcvar()`)
- `true_params`: list of true parameter values

Examples

```
sim <- simulate_dcvar(n_time = 100, rho_trajectory = rho_decreasing(100))
head(sim$Y_df)
plot(sim$true_params$rho, type = "l")
```

`simulate_dcvar_multilevel` *Simulate data from a multilevel copula VAR(1) model*

Description

Generates panel data with unit-specific VAR coefficients drawn from a population distribution and a global copula correlation. The simulator matches the fitted multilevel model support by leaving unit-level VAR matrices unconstrained; nonstationary draws are possible.

Usage

```
simulate_dcvar_multilevel(
  N = 40,
  n_time = 100,
  phi_bar = c(0.3, 0.1, 0.1, 0.3),
  tau_phi = c(0.1, 0.05, 0.05, 0.1),
  sigma = c(1, 1),
  rho = 0.3,
  margins = "normal",
  skew_direction = NULL,
  skew_params = NULL,
  burnin = 30,
  center = TRUE,
  seed = NULL
)
```

Arguments

`N`	Number of units.
`n_time`	Number of time points per unit.
`phi_bar`	Population mean for VAR coefficients (length-4 vector: phi11, phi12, phi21, phi22).
`tau_phi`	Population SD for each VAR coefficient (length-4 vector).
`sigma`	Innovation SDs (length-2 vector; used by normal dimensions).
`rho`	Global copula correlation.
`margins`	Marginal family. Either a single string applied to both variables, or a length-2 character vector for per-variable (mixed) margins, e.g. `c("normal", "exponential")`. Each entry is one of `"normal"` (default), `"exponential"`, `"skew_normal"`, or `"gamma"`.
`skew_direction`	Length-2 `1`/`-1` vector. Required whenever any dimension uses an `"exponential"` or `"gamma"` margin.
`skew_params`	Named list of margin-specific parameters: `alpha` (length-2 skew-normal shape) and/or `shape` (scalar gamma shape).
`burnin`	Number of burn-in observations to discard (default: 30).
`center`	Logical; person-mean center the data (default: `TRUE`).
`seed`	Random seed for reproducibility.

Value

A named list with:

- `data`: panel data frame with columns `id`, `time`, `y1`, `y2`
- `true_params`: list of true parameter values, including `phi_bar`, `tau_phi`, `sigma`, `rho`, `margins`, `skew_direction`, `skew_params`, and the per-unit VAR coefficients `Phi_mat` (an `N x 4` matrix) and `Phi_list` (a length-`N` list of `2 x 2` matrices)
- `person_means`: N x 2 matrix of person means (before centering)

`simulate_dcvar_sem` *Simulate data from a SEM copula VAR(1) model*

Description

Generates indicator-level time series data from a latent VAR(1) process with Gaussian copula dependence and a fixed measurement model.

Usage

```
simulate_dcvar_sem(
  n_time = 200,
  J = 3,
  lambda = rep(sqrt(0.8), 3),
  sigma_e = sqrt(0.2),
  Phi = matrix(c(0.5, 0.15, 0.15, 0.3), 2, 2),
  mu = c(0, 0),
  margins = "normal",
  sigma = c(1, 1),
  sigma_exp = c(1, 1),
  skew_direction = NULL,
  skew_params = NULL,
  rho = 0.3,
  burnin = 0,
  seed = NULL
)
```

Arguments

`n_time`	Number of time points.
`J`	Number of indicators per latent variable.
`lambda`	Numeric vector of length J with factor loadings.
`sigma_e`	Measurement error SD (scalar).
`Phi`	2x2 VAR coefficient matrix.
`mu`	Length-2 intercept vector.
`margins`	Latent-innovation marginal family. Either a single string (`"normal"` default or `"exponential"`), or a length-2 character vector for per-variable (mixed) margins, e.g. `c("normal", "gamma")`, where each entry is one of `"normal"`, `"exponential"`, `"skew_normal"`, or `"gamma"`.
`sigma`	Length-2 latent innovation SD vector for normal dimensions (also used by the normal dimensions of a mixed specification).
`sigma_exp`	Length-2 shifted-exponential scale vector for the single-family exponential path.
`skew_direction`	Integer vector of length 2 of `1`/`-1`. Required whenever any dimension uses an `"exponential"` or `"gamma"` margin.
`skew_params`	Named list of margin-specific parameters for mixed margins: `alpha` (length-2 skew-normal shape) and/or `shape` (scalar gamma shape).
`rho`	Copula correlation.
`burnin`	Retained for backward compatibility but ignored. Default `0` keeps the default simulation path aligned with the fitted SEM model, which conditions on `x_0 = 0` and treats the first returned state as observed rather than drawn after a burn-in period.
`seed`	Random seed for reproducibility.

Value

A named list with:

- `data`: data frame with columns `time`, `y1_1`, ..., `y1_J`, `y2_1`, ..., `y2_J`
- `true_params`: list of true parameter values
- `latent_states`: `n_time x 2` matrix of true latent states
- `innovations`: `n_time x 2` matrix of true innovations

`var_params` *Extract VAR(1) parameter summaries*

Description

Returns posterior summaries for the VAR parameters: intercepts (mu), coefficients (Phi), innovation SDs (sigma_eps), and sigma_omega (DC-VAR only).

Usage

```
var_params(object, ...)

## Default S3 method:
var_params(object, ...)

## S3 method for class 'dcvar_model_fit'
var_params(object, ...)

## S3 method for class 'dcvar_multilevel_fit'
var_params(object, ...)

## S3 method for class 'dcvar_sem_fit'
var_params(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Additional arguments (unused).

Details

For multilevel models, returns population-level parameters `phi_bar` (mean VAR coefficients), `tau_phi` (between-unit SDs), `sigma` (innovation SDs), and `rho` (copula correlation). These correspond to `Phi`, `sigma_eps`, and `rho` in single-level models.

Value

A named list of data frames with columns `variable`, `mean`, `sd`, `q2.5`, `q97.5`.

Index