Package 'cforecast'

May 8, 2026

Type Package

Title Conditional Forecasting and Scenario Analysis Using VAR Models

Version 0.1.0

Maintainer Tim Ginker <tim.ginker@gmail.com>

Description Provides tools for conducting scenario analysis in reduced-form vector autoregressive (VAR) models.
Implements a Kalman filtering framework to generate forecasts under path restrictions on selected variables. The package enables decomposition of conditional forecasts into variable-specific contributions, and extraction of observation weights. It also computes measures of overall and marginal variable importance to enhance the economic interpretation of forecast revisions. The framework is structurally agnostic and suited for policy analysis, stress testing, and macro-financial applications. The methodology is described in more detail in Caspi and Ginker (2026) <doi:10.13140/RG.2.2.25225.51040>.

Depends R (>= 3.5.0)

Imports BVAR, dplyr, FKF, miscTools, tibble, vars, utils, methods, wex

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

URL https://github.com/timginker/cforecast

BugReports https://github.com/timginker/cforecast/issues

NeedsCompilation no

Author Tim Ginker [aut, cre]

Repository CRAN

Date/Publication 2026-03-09 11:00:15 UTC

Contents

`cforecast` *Conditional Forecast with VAR or BVAR*

Description

Computes conditional forecasts using the Kalman filter and smoother methods proposed by Clarida and Coyle (1984) and Bańbura et al. (2015). Supports classical VAR models (`varest` from the **vars** package). Future implementations will also include Bayesian VAR models (`bvar` from the **BVAR** package). For now, for BVAR models the function allows analysis at the median of the parameter distribution.

Usage

```
cforecast(fit, cond_path, cond_var, horizon = NULL, p0 = 10000)
```

Arguments

`fit`	An object of class `varest` (from **vars**) or `bvar` (from **BVAR**)
`cond_path`	A numeric vector or matrix specifying the conditional path for the constrained variables
`cond_var`	A numeric vector indicating which columns of `y` are conditionally constrained
`horizon`	Optional forecast horizon (number of periods ahead). If `NULL`, it is inferred from the number of rows in `cond_path`
`p0`	Diagonal element of the initial state covariance matrix. Default is `1e4`

Value

A list with:

- `forecast` — Conditional forecast matrix (horizon × variables)
- `mse` — Forecast mean squared error (array: K × K × horizon)
- `fkf` — Output of the Kalman smoother (`FKF::fks`)
- `ss` — State space representation used in the forecast
- `cond_var` — Indices of constrained variables

- cond_path — The conditional path used
- horizon — Effective forecast horizon
- fit — The fitted VAR/BVAR object
- y — Full data matrix (historical + future with conditional constraints)

References

Clarida, R. and D. Coyle (1984). Conditional Projection by Means of Kalman Filtering. Carnegie-Rochester Conference Series on Public Policy, 20, 247–284.

Bańbura, M., Giannone, D., & Lenza, M. (2015). Conditional forecasts and scenario analysis with vector autoregressions for large cross-sections. *International Journal of Forecasting*, 31(3), 739–756.

Examples

```
library(vars)
data(Canada)
fit <- VAR(Canada, p = 2, type = "const")

# Define a conditional path: unemployment at 15 for 3 periods
cond_path <- matrix(rep(15, 3), ncol = 1)
cond_var <- 4  # 4th column is 'U'

conditional_forecast <- cforecast(fit = fit, cond_path = cond_path, cond_var = cond_var)
print(conditional_forecast$forecast)
```

---

cforecast_composition *Compute Conditional Forecast Composition*

Description

Compute Conditional Forecast Composition

Usage

```
cforecast_composition(x, target_var)
```

Arguments

x	cforecast result
target_var	target variable

Value

a data.frame with the decomposition

Examples

```
library(cforecast)
library(vars)
data(fred_macro)
# Fit a  VAR model
fit <- VAR(fred_macro[,-1], p = 2, type = "const")
# compute a conditional forecast given no change in the oil price DCOILWTICO in the next period
cond_path = 0
fct_constr <- cforecast(fit, cond_path = cond_path, cond_var = 5)
# compute forecast composition for PCEPILFE
infl_composition <- cforecast_composition(fct_constr, target_var =2)
```

DCOILWTICO_level *Quarterly Average WTI Crude Oil Price (Level)*

Description

Quarterly average spot price of West Texas Intermediate (WTI) crude oil at Cushing, Oklahoma.

Usage

```
DCOILWTICO_level
```

Format

A data frame with 161 observations and 2 variables:

date Date of observation (class `Date`)

DCOILWTICO Quarterly average oil price in U.S. dollars per barrel

Source

Federal Reserve Economic Data (FRED), Federal Reserve Bank of St. Louis. Data downloaded on 2026-01-27. https://fred.stlouisfed.org/

fct_weights *Conditional Forecast Variable Weights*

Description

Computes observation weights used in the Kalman smoother for a given target variable, showing the relative contribution of observed variables to the conditional forecast.

Usage

```
fct_weights(fit, cond_var, target_var, horizon, p0 = 10000)
```

Arguments

fit	An object of class varest (from **vars**) or bvar (from **BVAR**)
cond_var	A vector indicating which columns of 'y' are conditionally constrained
target_var	Column index of the target variable being forecast
horizon	Forecast horizon (number of future periods)
p0	diagonal element of the initial state covariance, with default of 1e4

Value

A list of weight matrices for each horizon step

Examples

```
library(cforecast)
library(vars)
data(fred_macro)
# Fit a  VAR model
fit <- VAR(fred_macro[,-1], p = 2, type = "const")
# compute observation weights for PCEPILFE given the conditioning on DCOILWTICO
fct_weights_fm <- fct_weights(fit = fit, cond_var = 5, target_var = 2, horizon = 1)
# note that only DCOILWTICO's weight is nonzero for h=1
```

fred_macro *U.S. Macroeconomic Dataset from FRED*

Description

Source: Federal Reserve Economic Data (FRED) https://fred.stlouisfed.org/

Usage

```
fred_macro
```

Format

An object of class `data.frame` with 158 rows and 6 columns.

Details

Data vintage: 2026-01-27.

All variables are obtained from the Federal Reserve Economic Data (FRED) database and converted to quarterly frequency. Observations are aligned to quarter-end dates to ensure consistency across series and compatibility with quarterly VAR estimation and forecasting.

Interest-rate and spread variables are averaged within each quarter, while price indices use end-of-period quarterly values.

Variable definitions:

GDPC1 Real Gross Domestic Product. Inflation-adjusted U.S. GDP, expressed as the quarter-to-quarter percent change.

PCEPILFE Core PCE Price Index. Personal Consumption Expenditures price index excluding food and energy ("core PCE"), expressed as the quarter-to-quarter percent change using end-of-period quarterly values.

FEDFUNDS Federal Funds Effective Rate. The effective federal funds rate, the primary short-term monetary policy rate in the United States. Constructed as the quarterly average and expressed in annualized percentage points (levels).

BAA10YM Corporate Credit Spread. The spread between Moody's seasoned Baa corporate bond yield and the 10-year U.S. Treasury constant maturity yield. Averaged within each quarter and expressed in percentage points.

DCOILWTICO Crude Oil Price (WTI). Spot price of West Texas Intermediate (WTI) crude oil at Cushing, Oklahoma. Quarterly average growth rate (quarter-to-quarter percent change).

Date convention: All quarterly observations are aligned to the last calendar day of the quarter to ensure temporal consistency across variables and compatibility with quarterly VAR and forecasting analysis.

`state_space_representation` *State Space Representation*

Description

Constructs the state space representation of a VAR model. Compatible with both classical VAR models (from the **vars** package) and Bayesian VAR models (from the **BVAR** package) for which the state space representation is computed at the median of the parameter distribution.

Usage

```
state_space_representation(fit, p0 = 10000)
```

Arguments

`fit`	An object of class `varest` (from **vars**) or `bvar` (from **BVAR**)
`p0`	diagonal element of the initial state covariance, with default of 1e4

Value

A list containing the state space system matrices:

- `Tt` — transition matrix
- `Zt` — observation matrix
- `GGt` — observation noise covariance
- `HHt` — process noise covariance
- `dt` — deterministic component (e.g., constant)
- `a0` — initial state vector
- `P0` — initial state covariance (diagonal, with default value `1e4`)

Author(s)

Tim Ginker

Examples

```
library(vars)
library(BVAR)
data(Canada)

# Classical VAR
fit_vars <- VAR(Canada, p = 2, type = "const")
ss_vars <- state_space_representation(fit_vars)

# Bayesian VAR (small example)
data <- fred_qd[, c("CPIAUCSL", "UNRATE", "FEDFUNDS")]
data <- fred_transform(data, codes = c(5, 5, 1), lag = 4)
fit_bvar <- bvar(data, lags = 2, n_draw = 1000L, n_burn = 200L, verbose = FALSE)
ss_bvar <- state_space_representation(fit_bvar)
```

`variable_importance_stat` *Compute Variable Importance for Stationary VAR*

Description

Computes the relative importance of each variable in a conditional forecast from a stationary VAR model. Supports both classical VAR models (from the **vars** package) and Bayesian VAR models (from the **BVAR** package) for which the computation is performed at the median of the parameter distribution.

Usage

```
variable_importance_stat(fit, cond_var, target_var, horizon)
```

Arguments

`fit`	An object of class `varest` (from **vars**) or `bvar` (from **BVAR**)
`cond_var`	A vector indicating which columns of `y` are conditionally constrained
`target_var`	Column index of the target variable being forecasted
`horizon`	Forecast horizon (number of future periods)

Value

A list containing two data frames:

- `variable_importance` — overall variable importance by horizon
- `marginal_variable_importance` — importance of future observations (marginal)

Examples

```
library(cforecast)
library(vars)
data(fred_macro)
# Fit a  VAR model
fit <- VAR(fred_macro[,-1], p = 2, type = "const")
# conditioning on the oil price, target variable is core inflation,
# horizon is set to 1 to speed up the computation
vim <- variable_importance_stat(fit = fit, cond_var = 5, target_var = 2, horizon = 1)
```

Index