Package: qardlr (via r-universe)

October 26, 2026

Type Package

Title Quantile Autoregressive Distributed Lag Model

Version 1.1.2

Date 2026-10-26

Description Implements the Quantile Autoregressive Distributed Lag (QARDL) model of Cho, Kim and Shin (2015) <doi:10.1016/j.jeconom.2015.05.003>. Estimates quantile-specific long-run (beta), short-run autoregressive (phi), and impact (gamma) parameters. Features include BIC-based automatic lag selection, Error Correction Model (ECM) parameterization, Wald tests for parameter constancy across quantiles, rolling/recursive QARDL estimation, Monte Carlo simulation, and publication-ready output tables.

License GPL-3

URL https://github.com/muhammedalkhalaf/qardlr

BugReports https://github.com/muhammedalkhalaf/qardlr/issues

Depends R (>= 3.5.0)

Imports quantreg (>= 5.95), stats, MASS

Suggests testthat (>= 3.0.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Config/testthat/edition 3

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

Date/Publication 2026-10-26 02:40:40 UTC

RemoteUrl https://github.com/muhammedalkhalaf/qardlr

RemoteRef HEAD

RemoteSha 2bb03fcec7970bc4ea5841e86a2f8a7b261daf5a

Contents

coef.qardl 2
plot.qardl_rolling 3
predict.qardl 3
print.qardl 4
print.qardl_mc 4
print.qardl_rolling 5
print.qardl_wald 5
print_bic_grid 6
qardl 6
qardl_bic_select 9
qardl_rolling 10
qardl_sim 12
qardl_simulate 13
qardl_table 15
qardl_wald 16
summary.qardl 17
vcov.qardl 18

Index **19**

`coef.qardl` *Coefficients Method for QARDL*

Description

Extract coefficients from a QARDL model.

Usage

```
## S3 method for class 'qardl'
coef(object, type = c("all", "beta", "phi", "gamma"), ...)
```

Arguments

`object`	An object of class `"qardl"`.
`type`	Character. Which coefficients to extract: `"beta"`, `"phi"`, `"gamma"`, or `"all"`. Default is `"all"`.
`...`	Additional arguments (unused).

Value

Matrix or list of coefficient matrices.

plot.qardl_rolling *Plot Rolling QARDL Results*

Description

Creates time series plots of rolling QARDL parameter estimates.

Usage

```
## S3 method for class 'qardl_rolling'
plot(x, which = c("beta", "phi", "gamma", "rho"), var = 1, tau_idx = NULL, ...)
```

Arguments

x	Object of class "qardl_rolling".
which	Character. Which parameter to plot: "beta", "phi", "gamma", or "rho". Default is "beta".
var	Integer or character. Which variable to plot (for beta/gamma). Default is 1.
tau_idx	Integer. Which quantile index to plot. Default is all.
...	Additional arguments passed to plot.

Value

Invisible NULL.

predict.qardl *Predict Method for QARDL*

Description

Generate predictions from a fitted QARDL model.

Usage

```
## S3 method for class 'qardl'
predict(object, newdata = NULL, tau = NULL, ...)
```

Arguments

object	An object of class "qardl".
newdata	Optional data frame for prediction. If NULL, uses the original data.
tau	Quantile(s) for prediction. Default uses all fitted quantiles.
...	Additional arguments (unused).

Value

Matrix of predicted quantiles.

`print.qardl` *Print QARDL Results*

Description

Print method for QARDL estimation results.

Usage

```
## S3 method for class 'qardl'
print(x, digits = 4, ...)
```

Arguments

`x`	An object of class `"qardl"`.
`digits`	Number of decimal places. Default is 4.
`...`	Additional arguments (unused).

Value

Invisible `x`.

`print.qardl_mc` *Print Monte Carlo Results*

Description

Print Monte Carlo Results

Usage

```
## S3 method for class 'qardl_mc'
print(x, digits = 4, ...)
```

Arguments

`x`	Object of class `"qardl_mc"`.
`digits`	Number of decimal places. Default is 4.
`...`	Additional arguments (unused).

Value

Invisible `x`.

print.qardl_rolling *Print Rolling QARDL Results*

Description

Print Rolling QARDL Results

Usage

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

Arguments

x	Object of class "qardl_rolling".
...	Additional arguments (unused).

Value

Invisible x.

print.qardl_wald *Print QARDL Wald Test Results*

Description

Print QARDL Wald Test Results

Usage

```
## S3 method for class 'qardl_wald'
print(x, digits = 4, ...)
```

Arguments

x	Object of class "qardl_wald".
digits	Number of decimal places. Default is 4.
...	Additional arguments (unused).

Value

Invisible x.

print_bic_grid *Print BIC Grid*

Description

Prints a formatted BIC grid for lag selection.

Usage

```
print_bic_grid(bic_result, digits = 3)
```

Arguments

bic_result	Result from qardl_bic_select.
digits	Number of decimal places. Default is 3.

Value

Invisible NULL. Called for side effect of printing.

Examples

```
data(qardl_sim)
y <- qardl_sim$y
X <- as.matrix(qardl_sim[, c("x1", "x2")])
bic_result <- qardl_bic_select(y, X, pmax = 4, qmax = 4)
print_bic_grid(bic_result)
```

qardl *Quantile Autoregressive Distributed Lag Model Estimation*

Description

Estimates the Quantile ARDL (QARDL) model of Cho, Kim & Shin (2015). The model estimates quantile-specific long-run equilibrium relationships and short-run dynamics between a dependent variable and covariates.

Usage

```
qardl(
  formula,
  data,
  tau = c(0.25, 0.5, 0.75),
  p = 0L,
  q = 0L,
```

```
  pmax = 7L,
  qmax = 7L,
  ecm = FALSE,
  constant = TRUE
)
```

Arguments

`formula`	A formula of the form `y ~ x1 + x2 + ...` where `y` is the dependent variable and `x1, x2, ...` are covariates.
`data`	A data frame containing the variables in the formula. The rows must form one contiguous time series without missing values; rows with `NA` are not dropped and raise an error.
`tau`	Numeric vector of quantiles to estimate. Must be in (0, 1). Default is `c(0.25, 0.50, 0.75)`.
`p`	Integer. AR lag order for the dependent variable. If 0, automatically selected via BIC. Default is 0.
`q`	Integer. Distributed lag order for covariates. If 0, automatically selected via BIC. Default is 0.
`pmax`	Integer. Maximum AR lag order for BIC selection. Default is 7.
`qmax`	Integer. Maximum DL lag order for BIC selection. Default is 7.
`ecm`	Logical. If `TRUE`, estimate Error Correction Model parameterization. Default is `FALSE`.
`constant`	Logical. If `TRUE`, include an intercept. Default is `TRUE`.

Details

The QARDL(p,q) model is specified as:

$$Q_{y_t}(\tau|\mathcal{F}_{t-1}) = c(\tau) + \sum_{i=1}^{p} \phi_i(\tau) y_{t-i} + \sum_{j=0}^{q-1} \gamma_j'(\tau) x_{t-j}$$

Long-run parameters are computed as:

$$\beta(\tau) = \frac{\sum_{j=0}^{q-1} \gamma_j(\tau)}{1 - \sum_{i=1}^{p} \phi_i(\tau)}$$

Standard errors and Wald tests follow Cho, Kim and Shin (2015); see `cks_covariance`.

The speed of adjustment (ECM coefficient) is:

$$\rho(\tau) = \sum_{i=1}^{p} \phi_i(\tau) - 1$$

Negative $\rho(\tau)$ indicates convergence to long-run equilibrium.

Value

An object of class `"qardl"` containing:

beta Long-run parameters matrix (k x ntau)

beta_se, beta_cov Standard errors and covariance matrices of beta (Cho, Kim and Shin, 2015, Theorem 2)

phi Short-run AR parameters matrix (p x ntau)

phi_se, phi_cov Standard errors and covariances of phi (Theorem 1 and equation 13)

gamma Level coefficients $\gamma(\tau) = \sum_j \gamma_j(\tau)$ (k x ntau), with standard errors `gamma_se`. The asymptotic covariance of $\hat{\gamma}(\tau)$ is $\hat{\beta}(\tau)\iota'\Pi(\tau)\iota\beta(\tau)'$, of rank one (Cho, Kim and Shin, 2015, Theorem 1(ii)), so the t statistics of the elements of $\gamma(\tau)$ are equal in absolute value to the t statistic of $\rho(\tau)$ and test the same hypothesis

gamma0, gamma0_se Coefficients on x_t (the impact of Δx_t in the ECM form) with quantile regression standard errors

rho, rho_se Speed of adjustment $\sum \phi_i(\tau) - 1$ and its standard error

cov_joint Joint covariance matrices of beta, phi, gamma and rho across quantiles (Theorems 3 and 4), used by `qardl_wald`

fhat Estimated error densities at the quantiles (Gaussian kernel, Bofinger bandwidth)

tau Vector of estimated quantiles

p AR lag order used

q Distributed lag order used

nobs Number of observations

k Number of covariates

call The matched call

formula The model formula

data The data used

qr_fits List of quantreg fit objects

bic_grid BIC grid if lag selection was performed

ecm Whether ECM parameterization was used

References

Cho, J.S., Kim, T.-H., & Shin, Y. (2015). Quantile cointegration in the autoregressive distributed-lag modeling framework. *Journal of Econometrics*, 188(1), 281-300. doi:10.1016/j.jeconom.2015.05.003

See Also

`qardl_rolling`, `qardl_simulate`, `qardl_wald`, `summary.qardl`

Examples

```
# Load example data
data(qardl_sim)

# Basic QARDL estimation with automatic lag selection
fit <- qardl(y ~ x1 + x2, data = qardl_sim, tau = c(0.25, 0.50, 0.75))
summary(fit)

# QARDL with specified lags
fit2 <- qardl(y ~ x1 + x2, data = qardl_sim, tau = c(0.1, 0.5, 0.9), p = 2, q = 2)
print(fit2)

# QARDL-ECM parameterization
fit_ecm <- qardl(y ~ x1 + x2, data = qardl_sim, tau = c(0.25, 0.50, 0.75), ecm = TRUE)
summary(fit_ecm)
```

---

qardl_bic_select *BIC-Based Lag Order Selection for QARDL*

Description

Automatically selects optimal lag orders (p, q) for the QARDL model using the Bayesian Information Criterion (BIC) evaluated at the median quantile (tau = 0.5).

Usage

```
qardl_bic_select(y, X, pmax = 7L, qmax = 7L, constant = TRUE)
```

Arguments

y	Numeric vector of dependent variable.
X	Matrix of covariates. y and X must form one contiguous series without missing values.
pmax	Integer. Maximum AR lag order to consider. Default is 7.
qmax	Integer. Maximum distributed lag order to consider. Default is 7.
constant	Logical. Include intercept. Default is TRUE.

Details

The BIC is computed using the Schwarz criterion at the median quantile:

$$BIC(p,q) = \log(\hat{\sigma}^2_{\tau=0.5}) + \frac{k_{pq}\log(n)}{n}$$

where k_{pq} is the number of parameters (p AR terms + q*k impact terms + constant) and $\hat{\sigma}^2$ is the estimated residual variance.

Value

A list containing:

p_opt Optimal AR lag order

q_opt Optimal distributed lag order

bic_grid Matrix of BIC values (pmax x qmax)

bic_min Minimum BIC value

References

Cho, J.S., Kim, T.-H., & Shin, Y. (2015). Quantile cointegration in the autoregressive distributed-lag modeling framework. *Journal of Econometrics*, 188(1), 281-300. doi:10.1016/j.jeconom.2015.05.003

See Also

qardl

Examples

```
data(qardl_sim)
y <- qardl_sim$y
X <- as.matrix(qardl_sim[, c("x1", "x2")])
bic_result <- qardl_bic_select(y, X, pmax = 5, qmax = 5)
print(bic_result$bic_grid)
```

---

qardl_rolling *Rolling Window QARDL Estimation*

Description

Performs rolling or recursive window QARDL estimation to assess parameter stability over time.

Usage

```
qardl_rolling(
  formula,
  data,
  tau = c(0.25, 0.5, 0.75),
  p = 1L,
  q = 1L,
  window = 0L,
  method = c("rolling", "recursive"),
  constant = TRUE
)
```

Arguments

formula	A formula of the form y ~ x1 + x2 + ...
data	A data frame containing the variables.
tau	Numeric vector of quantiles. Default is c(0.25, 0.50, 0.75).
p	Integer. AR lag order. Default is 1.
q	Integer. Distributed lag order. Default is 1.
window	Integer. Rolling window size. If 0, uses 10% of sample size.
method	Character. Either "rolling" (fixed window) or "recursive" (expanding window). Default is "rolling".
constant	Logical. Include intercept. Default is TRUE.

Details

Rolling window estimation helps detect structural breaks and assess parameter stability. The function estimates QARDL models on successive windows of data and tracks how parameters evolve over time.

For method = "rolling", a fixed window of size window is used. For method = "recursive", the window expands from window to the full sample.

Value

An object of class "qardl_rolling" containing:

beta 3D array of long-run parameters (k x ntau x nwindows)

phi 3D array of AR parameters (p x ntau x nwindows)

gamma 3D array of impact parameters (k x ntau x nwindows)

rho Matrix of ECM coefficients (nwindows x ntau)

wald_beta Matrix of beta constancy Wald statistics

wald_phi Matrix of phi constancy Wald statistics

wald_gamma Matrix of gamma constancy Wald statistics, one column per covariate

dates Vector of end dates for each window

window Window size used

method Method used ("rolling" or "recursive")

tau Vector of quantiles

References

Cho, J.S., Kim, T.-H., & Shin, Y. (2015). Quantile cointegration in the autoregressive distributed-lag modeling framework. *Journal of Econometrics*, 188(1), 281-300. doi:10.1016/j.jeconom.2015.05.003

See Also

qardl, plot.qardl_rolling

Examples

```
data(qardl_sim)

# Rolling estimation with 50-observation window
roll <- qardl_rolling(y ~ x1 + x2, data = qardl_sim,
                      tau = c(0.25, 0.50, 0.75), p = 2, q = 2, window = 50)
print(roll)

# Recursive estimation
recur <- qardl_rolling(y ~ x1 + x2, data = qardl_sim,
                       tau = c(0.50), p = 2, q = 2,
                       window = 50, method = "recursive")
print(recur)
```

`qardl_sim` *Simulated QARDL Dataset*

Description

A simulated dataset for demonstrating QARDL estimation. The data is generated from a QARDL(2,2) process with two covariates.

Usage

```
qardl_sim
```

Format

A data frame with 200 observations and 3 variables:

y Dependent variable generated from QARDL process

x1 First covariate (I(1) random walk)

x2 Second covariate (I(1) random walk)

Details

The data generating process follows:

$$y_t = 0.4y_{t-1} + 0.2y_{t-2} + 0.5x_{1t} + 0.3x_{2t} + u_t$$

where $u_t \sim N(0, 1)$ and x_{it} are independent random walks.

True parameters:

- $\phi_1 = 0.4$, $\phi_2 = 0.2$
- $\gamma_1 = 0.5$, $\gamma_2 = 0.3$
- $\beta_1 = 0.5/(1 - 0.6) = 1.25$, $\beta_2 = 0.3/(1 - 0.6) = 0.75$

References

Cho, J.S., Kim, T.-H., & Shin, Y. (2015). Quantile cointegration in the autoregressive distributed-lag modeling framework. *Journal of Econometrics*, 188(1), 281-300. doi:10.1016/j.jeconom.2015.05.003

Examples

```
data(qardl_sim)
head(qardl_sim)
summary(qardl_sim)

# Estimate QARDL model
fit <- qardl(y ~ x1 + x2, data = qardl_sim, tau = c(0.25, 0.50, 0.75), p = 2, q = 2)
summary(fit)
```

`qardl_simulate` *Monte Carlo Simulation for QARDL*

Description

Performs Monte Carlo simulation to assess the finite-sample properties of QARDL estimators under specified data generating processes.

Usage

```
qardl_simulate(
  nobs = 200L,
  reps = 1000L,
  tau = c(0.25, 0.5, 0.75),
  p = 1L,
  q = 1L,
  k = 1L,
  beta_true = NULL,
  phi_true = NULL,
  gamma_true = NULL,
  sigma_u = 1,
  sigma_x = 1,
  seed = NULL,
  parallel = FALSE,
  ncores = NULL
)
```

Arguments

`nobs`	Integer. Sample size for each simulation. Default is 200.
`reps`	Integer. Number of Monte Carlo replications. Default is 1000.
`tau`	Numeric vector of quantiles. Default is `c(0.25, 0.50, 0.75)`.

p	Integer. AR lag order. Default is 1.
q	Integer. Distributed lag order. Default is 1.
k	Integer. Number of covariates. Default is 1.
beta_true	Numeric vector. True long-run parameters (length k). Default is rep(1, k).
phi_true	Numeric vector. True AR parameters (length p). Default is rep(0.5, p).
gamma_true	Numeric vector. True impact parameters (length k). Default is rep(0.3, k).
sigma_u	Numeric. Standard deviation of the error term. Default is 1.
sigma_x	Numeric. Standard deviation of covariate innovations. Default is 1.
seed	Integer. Random seed for reproducibility. Default is NULL.
parallel	Logical. Use parallel processing. Default is FALSE.
ncores	Integer. Number of cores for parallel processing. Default is parallel::detectCores() - 1.

Details

The data generating process is:

$$y_t = \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{j=1}^{k} \gamma_j x_{jt} + u_t$$

where $u_t \sim N(0, \sigma_u^2)$ and x_{jt} follows a random walk with innovations $\sim N(0, \sigma_x^2)$.

Value

An object of class "qardl_mc" containing:

beta_sim Array of simulated beta estimates (k x ntau x reps)

phi_sim Array of simulated phi estimates (p x ntau x reps)

gamma_sim Array of simulated gamma estimates (k x ntau x reps)

beta_true True beta values

phi_true True phi values

gamma_true True gamma values

bias_beta Bias in beta estimates

rmse_beta RMSE of beta estimates

coverage_beta Empirical coverage of 95% CI for beta

reps Number of replications

nobs Sample size

tau Vector of quantiles

References

Cho, J.S., Kim, T.-H., & Shin, Y. (2015). Quantile cointegration in the autoregressive distributed-lag modeling framework. *Journal of Econometrics*, 188(1), 281-300. doi:10.1016/j.jeconom.2015.05.003

See Also

qardl, print.qardl_mc

Examples

```
# Small simulation for illustration
mc <- qardl_simulate(nobs = 100, reps = 50, tau = c(0.25, 0.50, 0.75),
                     p = 1, q = 1, k = 1, seed = 123)
print(mc)
```

---

qardl_table *Generate Publication-Ready QARDL Tables*

Description

Creates formatted tables suitable for academic publications from QARDL estimation results.

Usage

```
qardl_table(
  x,
  type = c("text", "latex", "html"),
  include = c("beta", "gamma"),
  stars = TRUE,
  digits = 3,
  caption = NULL,
  label = NULL
)
```

Arguments

x	An object of class "qardl".
type	Character. Type of table: "latex", "html", or "text". Default is "text".
include	Character vector. Which parameters to include: "beta", "phi", "gamma", "rho". Default is c("beta", "gamma").
stars	Logical. Include significance stars. Default is TRUE.
digits	Integer. Number of decimal places. Default is 3.
caption	Character. Table caption. Default is NULL.
label	Character. LaTeX label. Default is NULL.

Value

Character string containing the formatted table.

Examples

```
data(qardl_sim)
fit <- qardl(y ~ x1 + x2, data = qardl_sim, tau = c(0.25, 0.50, 0.75), p = 2, q = 2)
cat(qardl_table(fit, type = "text"))
```

`qardl_wald` *Wald Tests for QARDL Parameter Constancy*

Description

Performs Wald tests for parameter constancy across quantiles in a QARDL model. Tests whether parameters are equal across different quantile levels.

Usage

```
qardl_wald(
  object,
  type = c("all", "beta", "phi", "gamma", "rho"),
  pairwise = FALSE
)
```

Arguments

`object`	An object of class `"qardl"`.
`type`	Character string specifying which parameters to test: `"all"` (default), `"beta"` (long-run), `"phi"` (AR), `"gamma"` (short-run impact), or `"rho"` (ECM speed of adjustment).
`pairwise`	Logical. If `TRUE`, perform pairwise tests between adjacent quantiles. Default is `FALSE`.

Details

The Wald test statistic is computed as:

$$W = (R\hat{\theta} - r)'[R\hat{V}R']^{-1}(R\hat{\theta} - r) \sim \chi^2(q)$$

where R is a restriction matrix testing equality across quantiles, $\hat{\theta}$ is the vector of parameter estimates, and $\hat{V}$ is their joint covariance matrix across quantiles (Cho, Kim and Shin, 2015, Theorems 3 and 4).

Value

An object of class `"qardl_wald"` containing:

tests Data frame of test results with columns: test, statistic, df, pvalue

pairwise_tests Data frame of pairwise test results (if pairwise = TRUE)

type Type of test performed

tau Vector of quantiles

References

Cho, J.S., Kim, T.-H., & Shin, Y. (2015). Quantile cointegration in the autoregressive distributed-lag modeling framework. *Journal of Econometrics*, 188(1), 281-300. doi:10.1016/j.jeconom.2015.05.003

See Also

`qardl`, `print.qardl_wald`

Examples

```
data(qardl_sim)
fit <- qardl(y ~ x1 + x2, data = qardl_sim, tau = c(0.25, 0.50, 0.75), p = 2, q = 2)
wald_results <- qardl_wald(fit)
print(wald_results)

# Pairwise tests
wald_pairwise <- qardl_wald(fit, pairwise = TRUE)
print(wald_pairwise)
```

---

summary.qardl *Summary of QARDL Results*

Description

Provides a detailed summary of QARDL estimation results including parameter estimates, standard errors, t-statistics, p-values, and diagnostic tests.

Usage

```
## S3 method for class 'qardl'
summary(object, wald = TRUE, digits = 4, ...)
```

Arguments

`object`	An object of class `"qardl"`.
`wald`	Logical. Include Wald tests for parameter constancy. Default is `TRUE`.
`digits`	Number of decimal places. Default is 4.
`...`	Additional arguments (unused).

Value

An object of class `"summary.qardl"` (invisibly).

vcov.qardl *Variance-Covariance Method for QARDL*

Description

Extract variance-covariance matrices from a QARDL model.

Usage

```
## S3 method for class 'qardl'
vcov(object, type = c("all", "beta", "phi", "gamma"), ...)
```

Arguments

object	An object of class "qardl".
type	Character. Which covariance to extract: "beta", "phi", "gamma", or "all". Default is "all".
...	Additional arguments (unused).

Value

Array or list of covariance arrays.

Index

∗ **datasets**
qardl_sim, 12

cks_covariance, *7*
coef.qardl, 2

plot.qardl_rolling, 3, *11*
predict.qardl, 3
print.qardl, 4
print.qardl_mc, 4, *15*
print.qardl_rolling, 5
print.qardl_wald, 5, *17*
print_bic_grid, 6

qardl, 6, *10, 11*, *15*, *17*
qardl_bic_select, 9
qardl_rolling, *8*, 10
qardl_sim, 12
qardl_simulate, *8*, 13
qardl_table, 15
qardl_wald, *8*, 16

summary.qardl, *8*, 17

vcov.qardl, 18