Package ‘csdm’

March 23, 2026

Title Cross-Sectional Dependence Models

Version 1.0.1

Depends R (>= 4.0.0)

Imports MASS, Rdpack

RdMacros Rdpack

Suggests testthat (>= 3.0.0), knitr, rmarkdown, kableExtra, xtsum, spelling

URL https://github.com/Macosso/csdm

BugReports https://github.com/Macosso/csdm/issues

Description Provides estimators and utilities for large panel-data models with cross-sectional dependence, including mean group (MG), common correlated effects (CCE) and dynamic CCE (DCCE) estimators, and cross-sectionally augmented ARDL (CS-ARDL) specifications, plus related inference and diagnostics.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Config/testthat/edition 3

VignetteBuilder knitr

Language en-US

NeedsCompilation no

Author Joao Claudio Macosso [aut, cre] (ORCID: <https://orcid.org/0009-0006-5051-9312>)

Maintainer Joao Claudio Macosso <joaoclaudiomacosso@gmail.com>

Repository CRAN

Date/Publication 2026-03-23 21:50:02 UTC

Contents

`cd_test` *Cross-sectional dependence (CD) tests for panel residuals*

Description

Computes Pesaran CD, CDw, CDw+, and CD* tests for cross-sectional dependence in panel residuals. The implementation supports residual matrices or fitted `csdm_fit` objects and provides consistent handling of unbalanced panels.

Usage

```
cd_test(object, ...)

## Default S3 method:
cd_test(
  object,
  type = c("CD", "CDw", "CDw+", "CDstar", "all"),
  n_pc = 4L,
  seed = NULL,
  min_overlap = 2L,
  na.action = c("drop.incomplete.times", "pairwise"),
  ...
)

## S3 method for class 'csdm_fit'
cd_test(
  object,
  type = c("CD", "CDw", "CDw+", "CDstar", "all"),
  n_pc = 4L,
    seed = NULL,
    min_overlap = 2L,
    na.action = c("drop.incomplete.times", "pairwise"),
    ...
)

## S3 method for class 'cd_test'
print(x, digits = 3, ...)
```

Arguments

`object`	A `csdm_fit` model object or a numeric matrix of residuals (N x T).
`...`	Additional arguments passed to methods.
`type`	Which test(s) to compute: one of `"CD"`, `"CDw"`, `"CDw+"`, `"CDstar"`, or `"all"` (default: `"CD"`).
`n_pc`	Number of principal components for CD* (default 4).
`seed`	Integer seed for weight draws in CDw/CDw+ (default NULL = no seed set).
`min_overlap`	Minimum number of overlapping time periods required for a unit pair to be included in CD/CDw/CDw+ (default 2).
`na.action`	How to handle missing data: `"drop.incomplete.times"` (default) removes time periods with any missing observations to create a balanced panel for CD*; `"pairwise"` uses pairwise correlations for CD/CDw/CDw+ and warns for CD*.
`x`	An object of class `cd_test`.
`digits`	Number of digits to print (default 3).

Details

Notation:

Let E be the residual matrix with N cross-sectional units and T time periods. For each unit pair (i, j), let T_{ij} be the number of overlapping time periods and ρ_{ij} the pairwise correlation.

Test statistics:

CD (Pesaran, 2015)

$$CD = \sqrt{\frac{2}{N(N-1)}} \sum_{i<j} \sqrt{T_{ij}}\, \rho_{ij}$$

CDw (Juodis and Reese, 2021) Random sign flips $w_i \in \{-1, 1\}$ are applied to residuals before computing correlations. The statistic is CD applied to the sign-flipped data.

CDw+ (Fan, Liao, and Yao, 2015) Power enhancement adds a sparse thresholding term to CDw. The threshold is

$$c_N = \sqrt{\frac{2\log(N)}{T}}$$

and the power term sums $\sqrt{T_{ij}}|\rho_{ij}|$ for pairs exceeding the threshold.

CD* (Pesaran and Xie, 2021) CD is computed on residuals after removing `n_pc` principal components from E. This provides a bias-corrected test under multifactor errors.

Missing data and balance:

CD, CDw, CDw+ Always use pairwise-complete observations. Each pairwise correlation uses available overlaps.

CD* Requires a balanced panel. By default, `na.action = "drop.incomplete.times"` removes any time period with missing observations. With `na.action = "pairwise"`, CD* returns `NA` and a warning when missing values are present.

Value

An object of class `cd_test` with fields `tests`, `type`, `N`, `T`, `na.action`, and `call`. The `tests` list contains one or more test results, each with `statistic` and `p.value`.

References

Pesaran MH (2015). "Testing weak cross-sectional dependence in large panels." *Econometric Reviews*, **34**(6-10), 1089–1117.

Pesaran MH (2021). "General diagnostic tests for cross-sectional dependence in panels." *Empirical Economics*, **60**(1), 13–50.

Juodis A, Reese S (2021). "The incidental parameters problem in testing for remaining cross-sectional correlation." *Journal of Business and Economic Statistics*, **40**(3), 1191–1203.

Fan J, Liao Y, Yao J (2015). "Power Enhancement in High-Dimensional Cross-Section Tests." *Econometrica*, **83**(4), 1497–1541.

Pesaran MH, Xie Y (2021). "A bias-corrected CD test for error cross-sectional dependence in panel models." *Econometric Reviews*, **41**(6), 649–677.

Examples

```
# Simulate independent and dependent panels
set.seed(1)
E_indep <- matrix(rnorm(100), nrow = 10)
E_dep <- matrix(rnorm(10), nrow = 10, ncol = 10, byrow = TRUE)

# Compute all tests
cd_test(E_indep, type = "all")
cd_test(E_dep, type = "all")

# Specific test with parameters
cd_test(E_indep, type = "CDstar", n_pc = 2)

# From a fitted csdm model
data(PWT_60_07, package = "csdm")
df <- PWT_60_07
ids <- unique(df$id)[1:10]
df_small <- df[df$id %in% ids & df$year >= 1970, ]
fit <- csdm(
  log_rgdpo ~ log_hc + log_ck + log_ngd,
  data = df_small,
  id = "id",
  time = "year",
  model = "cce",
  csa = csdm_csa(vars = c("log_rgdpo", "log_hc", "log_ck", "log_ngd"))
)
cd_test(fit, type = "all")
```

coef.csdm_fit *Extract model coefficients from a fitted csdm model*

Description

Returns estimated mean-group coefficients from a `csdm_fit` object. For `model = "cs_ardl"`, the returned vector includes short-run mean-group coefficients, the adjustment coefficient (named `lr_<y>`), and long-run coefficients when available.

Usage

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

Arguments

`object`	A fitted object of class `csdm_fit`.
`...`	Currently unused.

Value

A named numeric vector of estimated coefficients.

See Also

`summary.csdm_fit()`, `vcov.csdm_fit()`

csdm *Panel Model Estimation with Cross-Sectional Dependence*

Description

Estimate heterogeneous panel data models with optional cross-sectional augmentation and dynamic structure. The interface supports Mean Group (MG), Common Correlated Effects (CCE), Dynamic CCE (DCCE), and Cross-Sectionally Augmented ARDL (CS-ARDL) estimators with a consistent specification workflow for cross-sectional averages, lag structure, and variance-covariance estimation.

Usage

```
csdm(
  formula,
  data,
  id,
  time,
  model = c("mg", "cce", "dcce", "cs_ardl", "cs_ecm", "cs_dl"),
  csa = csdm_csa(),
  lr = csdm_lr(),
  pooled = csdm_pooled(),
  trend = c("none", "unit", "pooled"),
  fullsample = FALSE,
  mgmissing = FALSE,
  vcov = csdm_vcov(),
  ...
)
```

Arguments

`formula` Model formula of the form `y ~ x1 + x2`.

`data` A `data.frame` (or `plm::pdata.frame`) containing the variables in `formula`.

`id, time` Column names (strings) for the unit and time indexes. If `data` is a `pdata.frame`, these are taken from its index and the provided values are ignored.

`model` Estimator to fit. One of `"mg"`, `"cce"`, `"dcce"`, or `"cs_ardl"`.

`csa` Cross-sectional-average specification, created by `csdm_csa()`.

`lr` Long-run or dynamic specification, created by `csdm_lr()`.

`pooled` Pooled specification (reserved for future use), created by `csdm_pooled()`.

`trend` One of `"none"` or `"unit"` (adds a linear unit trend). `"pooled"` is reserved and not implemented.

`fullsample` Logical; reserved for future extensions.

`mgmissing` Logical; reserved for future extensions.

`vcov` Variance-covariance specification, created by `csdm_vcov()`.

`...` Reserved for future extensions.

Details

Let $i = 1, \ldots, N$ index cross-sectional units and $t = 1, \ldots, T$ index time. A baseline heterogeneous panel model is

$$y_{it} = \alpha_i + \beta_i^T x_{it} + u_{it}.$$

Here α_i is a unit-specific intercept, x_{it} is a vector of regressors, β_i is a vector of unit-specific slopes, and u_{it} is an error term that may exhibit cross-sectional dependence.

Cross-sectional averages are specified through `csdm_csa()` and dynamic or long-run structure is specified through `csdm_lr()`. This keeps the model interface consistent across estimators while allowing the degree of cross-sectional augmentation and lag structure to vary by application.

Implemented estimators

MG (Pesaran and Smith, 1995)

The Mean Group estimator fits separate regressions for each unit and averages the resulting coefficients:

$$\hat{\beta}_{MG} = \frac{1}{N}\sum_{i=1}^{N}\hat{\beta}_i.$$

This estimator accommodates slope heterogeneity but does not explicitly model cross-sectional dependence.

CCE (Pesaran, 2006)

Regressions are augmented with cross-sectional averages to proxy unobserved common factors:

$$y_{it} = \alpha_i + \beta_i^T x_{it} + \gamma_i^T \bar{z}_t + v_{it}.$$

A common choice is

$$\bar{z}_t = (\bar{y}_t, \bar{x}_t),$$

with

$$\bar{x}_t = \frac{1}{N}\sum_{i=1}^{N} x_{it}, \qquad \bar{y}_t = \frac{1}{N}\sum_{i=1}^{N} y_{it}.$$

More generally, $\bar{z}_t$ collects the cross-sectional averages specified in `csa`.

DCCE (Chudik and Pesaran, 2015)

Dynamic CCE extends CCE by allowing lagged dependent variables and lagged cross-sectional averages:

$$y_{it} = \alpha_i + \sum_{p=1}^{P}\phi_{ip}y_{i,t-p} + \beta_i^T x_{it} + \sum_{q=0}^{Q}\delta_{iq}^T \bar{z}_{t-q} + e_{it}.$$

In the package implementation, lagged dependent variables and distributed lags of regressors are controlled through `lr`, while contemporaneous and lagged cross-sectional averages are controlled through `csa`.

CS-ARDL (Chudik and Pesaran, 2015)

In the package implementation, `model = "cs_ardl"` is obtained by first estimating a cross-sectionally augmented ARDL-style regression in levels, using the same dynamic specification as `model = "dcce"`, and then transforming the unit-specific coefficients into adjustment and long-run parameters.

The underlying unit-level regression is of the form

$$y_{it} = \alpha_i + \sum_{p=1}^{P} \phi_{ip} y_{i,t-p} + \sum_{q=0}^{Q} \beta_{iq}^T x_{i,t-q} + \sum_{s=0}^{S} \omega_{is}^T \bar{z}_{t-s} + e_{it}.$$

From this dynamic specification, the package recovers the implied error-correction form

$$\Delta y_{it} = \alpha_i + \varphi_i \left(y_{i,t-1} - \theta_i^T x_{i,t-1} \right) + \sum_{j=1}^{P-1} \lambda_{ij} \Delta y_{i,t-j} + \sum_{j=0}^{Q-1} \psi_{ij}^T \Delta x_{i,t-j} + \sum_{s=0}^{S} \tilde{\omega}_{is}^T \bar{z}_{t-s} + e_{it},$$

where φ_i is the adjustment coefficient and θ_i is the implied long-run relationship. In the current implementation, these quantities are computed from the estimated lag polynomials rather than from a direct ECM regression.

Identification and assumptions

MG requires sufficient time-series variation within each unit.

CCE relies on cross-sectional averages acting as proxies for latent common factors, together with adequate cross-sectional and time dimensions.

DCCE additionally requires enough time periods to support lagged dependent variables, distributed lags, and lagged cross-sectional averages.

CS-ARDL requires sufficient time length for the distributed-lag structure and is intended for applications where both short-run dynamics and long-run relationships are of interest in the presence of common factors.

Value

An object of class `csdm_fit` containing estimated coefficients, residuals, variance-covariance estimates, model metadata, and diagnostics. Use `summary()`, `coef()`, `residuals()`, `vcov()`, and `cd_test()` to access standard outputs.

References

Pesaran MH, Smith R (1995). "Estimating long-run relationships from dynamic heterogeneous panels." *Journal of Econometrics*, **68**(1), 79–113.

Pesaran MH (2006). "Estimation and inference in large heterogeneous panels with multifactor error structure." *Econometrica*, **74**(4), 967–1012.

Chudik A, Pesaran MH (2015). "Common correlated effects estimation of heterogeneous dynamic panel data models with weakly exogenous regressors." *Journal of Econometrics*, **188**(2), 393–420.

Examples

```
library(csdm)
data(PWT_60_07, package = "csdm")
df <- PWT_60_07

# Keep examples fast but fully runnable
keep_ids <- unique(df$id)[1:10]
```

```
df_small <- df[df$id %in% keep_ids & df$year >= 1970, ]

# Mean Group (MG)
mg <- csdm(
  log_rgdpo ~ log_hc + log_ck + log_ngd,
  data = df_small, id = "id", time = "year", model = "mg"
)
summary(mg)

# Common Correlated Effects (CCE)
cce <- csdm(
  log_rgdpo ~ log_hc + log_ck + log_ngd,
  data = df_small, id = "id", time = "year", model = "cce",
  csa = csdm_csa(vars = c("log_rgdpo", "log_hc", "log_ck", "log_ngd"))
)
summary(cce)

# Dynamic CCE (DCCE)
dcce <- csdm(
  log_rgdpo ~ log_hc + log_ck + log_ngd,
  data = df_small, id = "id", time = "year", model = "dcce",
  csa = csdm_csa(vars = c("log_rgdpo", "log_hc", "log_ck", "log_ngd"), lags = 3),
  lr = csdm_lr(type = "ardl", ylags = 1, xdlags = 0)
)
summary(dcce)

# CS-ARDL
cs_ardl <- csdm(
  log_rgdpo ~ log_hc + log_ck + log_ngd,
  data = df_small, id = "id", time = "year", model = "cs_ardl",
  csa = csdm_csa(vars = c("log_rgdpo", "log_hc", "log_ck", "log_ngd"), lags = 3),
  lr = csdm_lr(type = "ardl", ylags = 1, xdlags = 0)
)
summary(cs_ardl)
```

csdm_csa *Specification: Cross-sectional averages (CSA)*

Description

Specification: Cross-sectional averages (CSA)

Usage

```
csdm_csa(
  vars = "_all",
  lags = 0,
  scope = c("estimation", "global", "cluster"),
  cluster = NULL
)
```

Arguments

vars Character. One of "_all", "_none", or a character vector of variable names.

lags Integer. Either a scalar integer >= 0 applied to all CSA variables, or a named integer vector giving per-variable maximum lags.

scope Character vector. One or more of c("estimation","global","cluster").

cluster Reserved for future use.

Value

A spec object (list) used by csdm().

Examples

```
# Cross-sectional averages (CSA) configuration for DCCE
csa <- csdm_csa(
  vars = c("log_rgdpo", "log_hc", "log_ck", "log_ngd"),
  lags = 3
)
csa
```

csdm_lr *Specification: Long-run configuration*

Description

Specification: Long-run configuration

Usage

```
csdm_lr(
  vars = NULL,
  type = c("none", "ecm", "ardl", "csdl"),
  ylags = 0,
  xdlags = 0,
  options = list()
)
```

Arguments

vars Reserved for future use.

type One of c("none","ecm","ardl","csdl").

ylags Integer >= 0. Within-unit lags of the dependent variable to include when supported by the chosen model/type.

xdlags Integer >= 0. Scalar distributed lags to apply to each RHS regressor when supported by the chosen model/type.

options Reserved for future use.

Value

A spec object (list) used by csdm().

Examples

```
# Long-run / dynamic configuration (ARDL-style lags)
lr <- csdm_lr(type = "ardl", ylags = 1)
lr

# Minimal end-to-end DCCE example (kept small for speed)
data(PWT_60_07, package = "csdm")
df <- PWT_60_07
keep_ids <- unique(df$id)[1:10]
df_small <- df[df$id %in% keep_ids & df$year >= 1970, ]
fit <- csdm(
  log_rgdpo ~ log_hc + log_ck + log_ngd,
  data = df_small,
  id = "id",
  time = "year",
  model = "dcce",
  csa = csdm_csa(vars = c("log_rgdpo", "log_hc", "log_ck", "log_ngd"), lags = 3),
  lr = csdm_lr(type = "ardl", ylags = 1)
)
summary(fit)
```

`csdm_pooled` *Specification: Pooled constraints (stub)*

Description

Specification: Pooled constraints (stub)

Usage

```
csdm_pooled(vars = NULL, constant = FALSE, trend = FALSE)
```

Arguments

`vars`	Reserved for future use.
`constant`	Logical; pooled constant.
`trend`	Logical; pooled trend.

Value

A spec object (list) used by csdm().

`csdm_vcov` *Specification: Variance-covariance for MG output (stub)*

Description

Specification: Variance-covariance for MG output (stub)

Usage

```
csdm_vcov(type = c("mg", "np", "nw", "wpn", "ols"), ...)
```

Arguments

`type`	One of c("mg","np","nw","wpn","ols").
`...`	Reserved for future use.

Value

A spec object (list) used by csdm().

`predict.csdm_fit` *Predict method for csdm models*

Description

Produces fitted values (index `"xb"`) when available, or returns model residuals. Prediction on new data is not yet implemented.

Usage

```
## S3 method for class 'csdm_fit'
predict(object, newdata = NULL, type = c("xb", "residuals"), ...)
```

Arguments

`object`	A fitted object of class `csdm_fit`.
`newdata`	Optional new data (not yet supported).
`type`	One of `"xb"` for fitted values or `"residuals"`.
`...`	Currently unused.

Value

A numeric matrix of fitted values or residuals, depending on `type`.

See Also

`residuals.csdm_fit()`, `summary.csdm_fit()`

print.csdm_fit *Compact print method for fitted csdm models*

Description

Prints a concise overview of a fitted `csdm_fit` object, including the model type, formula, panel dimensions, and a coefficient table with standard errors when available.

Usage

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

Arguments

`x`	A fitted object of class `csdm_fit`.
`digits`	Number of printed digits.
`...`	Currently unused.

Value

Invisibly returns `x`.

See Also

`summary.csdm_fit()`, `coef.csdm_fit()`, `residuals.csdm_fit()`

print.summary.csdm_fit *Print method for csdm summary objects*

Description

Formats and prints a `summary.csdm_fit` object. Output adapts to model type and includes coefficient tables, selected goodness-of-fit diagnostics, and compact model metadata.

Usage

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

Arguments

`x`	A `summary.csdm_fit` object.
`digits`	Number of digits to print.
`...`	Further arguments passed to methods.

Details

The printout includes classic Pesaran CD diagnostics from the summary object. For a full CD diagnostic panel (CD, CDw, CDw+, CD*), use `cd_test()` on the fitted model.

Value

Invisibly returns x.

See Also

`summary.csdm_fit()`, `cd_test()`

PWT_60_07 *Penn World Tables panel (93 countries, 1960-2007)*

Description

A panel of 93 countries (unit id) observed annually over 1960-2007 (time/year), with the log-transformed variables used in xtdcce2-style examples.

Usage

```
PWT_60_07
```

Format

A data frame with 4464 rows and 6 variables:

id Unit identifier (country id).

year Time identifier (year, 1960-2007).

log_rgdpo Log real GDP (output).

log_hc Log human capital index.

log_ck Log capital stock.

log_ngd Log (net) government debt (or similar), used as a covariate/control.

Source

Penn World Table (PWT). This dataset is included as a small, convenient panel for examples and tests.

residuals.csdm_fit *Extract residual matrix from a fitted csdm model*

Description

Returns residuals as an NxT matrix (rows are units, columns are time). This method is designed for panel diagnostics and downstream tools such as cd_test().

Usage

```
## S3 method for class 'csdm_fit'
residuals(object, type = c("e", "u"), ...)
```

Arguments

object	A fitted object of class csdm_fit.
type	Residual type. Currently only "e" is implemented.
...	Currently unused.

Value

A numeric matrix of residuals with dimensions NxT.

See Also

get_residuals(), cd_test(), predict.csdm_fit()

summary.csdm_fit *Summarize csdm model estimation results*

Description

Computes post-estimation summaries for csdm_fit objects, including mean-group coefficient inference, model-level diagnostics, and model-specific summary tables (for example, short-run and long-run blocks for CS-ARDL).

Usage

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

Arguments

object	A fitted model object of class csdm_fit.
digits	Number of digits to print.
...	Further arguments passed to methods.

Details

Reported inference:

For each coefficient $\hat{\beta}_k$, the summary reports standard errors, z-statistics, and two-sided normal-approximation p-values:

$$z_k = \frac{\hat{\beta}_k}{\mathrm{se}(\hat{\beta}_k)}, \qquad p_k = 2\{1 - \Phi(|z_k|)\}.$$

Diagnostics:

The printed summary shows the classic Pesaran CD diagnostic by default. Extended diagnostics (CDw, CDw+, CD*) are available through `cd_test()`.

Value

An object of class `summary.csdm_fit` with core metadata (call/formula/model/N/T), coefficient tables, fit statistics, and model-specific components for printing and downstream inspection.

See Also

`print.summary.csdm_fit()`, `cd_test()`, `coef.csdm_fit()`, `vcov.csdm_fit()`

Examples

```
data(PWT_60_07, package = "csdm")
df <- PWT_60_07
ids <- unique(df$id)[1:10]
df_small <- df[df$id %in% ids & df$year >= 1970, ]
fit <- csdm(
  log_rgdpo ~ log_hc + log_ck + log_ngd,
  data = df_small,
  id = "id",
  time = "year",
  model = "cce",
  csa = csdm_csa(vars = c("log_rgdpo", "log_hc", "log_ck", "log_ngd"))
)
s <- summary(fit)
s
```

`vcov.csdm_fit` *Extract coefficient covariance matrix from a fitted csdm model*

Description

Extract coefficient covariance matrix from a fitted csdm model

Usage

```
## S3 method for class 'csdm_fit'
vcov(object, ...)
```

Arguments

`object`	A fitted object of class `csdm_fit`.
`...`	Currently unused.

Value

A numeric variance-covariance matrix aligned with `coef(object)` for models where this is available.

See Also

`coef.csdm_fit()`, `summary.csdm_fit()`

Index