Package: ApproxOp (via r-universe)

October 17, 2026

Type Package

Title A Unified Framework for Positive Linear Approximation Operators

Version 0.1.0

Description Provides a unified computational framework for defining, evaluating, analyzing, comparing, and visualizing positive linear approximation operators. The package supports classical and generalized discrete operators, ordinary and central moments, numerical moment verification, approximation error analysis, convergence studies, parameter studies, and user-defined operators. The implemented framework is related to classical and generalized approximation operators described in Szász (1950) <doi:10.6028/JRES.045.024>, Sucu and Varma (2015) <doi:10.1016/j.amc.2015.08.077>, and Aliaga and Aslan (2026) <doi:10.1002/mma.70924>.

License GPL-3

Encoding UTF-8

LazyData true

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

URL https://github.com/edmond-aliaga/ApproxOp

BugReports https://github.com/edmond-aliaga/ApproxOp/issues

Repository https://edmond-aliaga.r-universe.dev

Date/Publication 2026-10-06 21:36:32 UTC

RemoteUrl https://github.com/edmond-aliaga/approxop

RemoteRef HEAD

RemoteSha 5c42030b6cbcd991609307ca3dfa0f2bd7d528c7

Contents

approx_error . . . 2
approx_operator . . . 3
approximate . . . 4
approximation_data . . . 4
check_operator . . . 5
compare_operators . . . 6
convergence . . . 6
error_table . . . 7
moments . . . 8
parameter_scan . . . 8
parameter_scan_2d . . . 9
plot.approx_convergence . . . 11
plot.approx_operator . . . 12
plot.approx_operator_comparison . . . 13
plot.approx_parameter_scan . . . 15
plot.approx_parameter_scan_2d . . . 16
print.approx_error_table . . . 18
print.approx_operator . . . 18
print.approx_operator_check . . . 19
register_operator . . . 19
verify_moments . . . 20

Index **22**

`approx_error` *Compute Approximation Errors*

Description

Computes pointwise approximation errors for an approximation operator at one or more evaluation points.

Usage

```
approx_error(operator, f, x)
```

Arguments

`operator`	An object of class `"approx_operator"`.
`f`	Function to be approximated.
`x`	Numeric vector of evaluation points.

Details

For an operator L_n and a function f, the signed error is

L_n(f; x) - f(x),

while the absolute and relative errors are

|L_n(f; x) - f(x)|

and

|L_n(f; x) - f(x)| / |f(x)|,

respectively. If f(x) = 0, the relative error is returned as NA.

Value

An object of class `"approx_error"` and `"data.frame"` containing the evaluation points, exact function values, approximated values, signed errors, absolute errors, and relative errors.

`approx_operator` *Create an Approximation Operator*

Description

Creates an approximation operator object from a supported operator family and, when applicable, a supported subfamily.

Usage

```
approx_operator(
  family,
  n,
  subfamily = NULL,
  variant = "discrete",
  params = list(),
  control = list()
)
```

Arguments

`family`	Character string specifying the operator family.
`n`	Positive integer specifying the approximation parameter.
`subfamily`	Optional character string specifying the operator subfamily. The default is `NULL`. For families that provide a general formulation, `NULL` represents the general family.
`variant`	Character string specifying the operator variant.
`params`	A list containing family- or subfamily-specific parameters.
`control`	A list containing numerical control parameters.

Value

An object of class "approx_operator".

---

approximate — *Evaluate an Approximation Operator*

Description

Evaluates a positive linear approximation operator for a given function at one or more points.

Usage

```
approximate(operator, f, x)
```

Arguments

operator	An object of class "approx_operator".
f	Function to be approximated.
x	Numeric vector of evaluation points.

Value

A numeric vector containing the approximated values.

---

approximation_data — *Create Approximation Data*

Description

Creates numerical data for studying and plotting the approximation of a function by an approximation operator for one or more values of n.

Usage

```
approximation_data(operator, f, x, n = NULL)
```

Arguments

operator	An object of class "approx_operator".
f	Function to be approximated.
x	Numeric vector of evaluation points.
n	Optional vector of positive integers. If NULL, the value of n stored in operator is used.

Details

For each evaluation point x and each value of n, the function computes the exact value f(x), the approximation L_n(f; x), the signed error, the absolute error, and the relative error.

Value

An object of class "approximation_data" and "data.frame" containing the columns x, n, exact, approximation, signed_error, absolute_error, and relative_error.

---

check_operator *Check an Approximation Operator*

Description

Performs a collection of structural and numerical diagnostic checks for an approximation operator.

Usage

```
check_operator(operator, x = NULL, tolerance = 1e-10, check_positivity = TRUE)
```

Arguments

operator	An object of class "approx_operator".
x	Optional numeric vector of diagnostic evaluation points. If NULL, suitable points are selected automatically from the operator domain.
tolerance	Positive numeric value used for numerical comparisons.
check_positivity	Logical. If TRUE, simple numerical positivity checks are performed.

Details

The function checks whether the operator can be evaluated successfully on a set of points, whether the returned values are finite, whether constants are reproduced within a specified tolerance, and optionally whether the operator appears to preserve non-negativity for a collection of simple non-negative test functions.

This function is intended as a general diagnostic tool. Detailed verification of ordinary or central moments is provided separately by verify_moments().

Value

An object of class "approx_operator_check" containing a data frame of diagnostic checks and an overall status.

compare_operators *Compare Approximation Operators*

Description

Compares two or more approximation operators numerically for the same function, evaluation grid, and one or more values of n.

Usage

```
compare_operators(operators, f, grid, n = NULL)
```

Arguments

operators	A list of objects of class "approx_operator". Named lists are recommended because the names are used as operator labels in the returned table.
f	Function to be approximated.
grid	Numeric vector of evaluation points.
n	Optional vector of positive integers. If NULL, each operator is evaluated using its own stored value of n.

Details

For each operator and each value of n, the function computes the maximum absolute error, mean absolute error (MAE), and root mean squared error (RMSE).

The comparison is performed on a common evaluation grid. Therefore, every point in grid must belong to the domain of every operator included in the comparison.

Value

An object of class "approx_operator_comparison" and "data.frame". The returned data frame contains the operator label, family, subfamily, variant, n, maximum absolute error, MAE, and RMSE.

convergence *Study Numerical Convergence*

Description

Computes numerical error measures for an approximation operator over a grid of evaluation points and for one or more values of n.

Usage

```
convergence(operator, f, grid, n)
```

Arguments

`operator`	An object of class `"approx_operator"`.
`f`	Function to be approximated.
`grid`	Numeric vector of evaluation points.
`n`	Vector of positive integers.

Details

For each value of `n`, the function computes the maximum absolute error, mean absolute error (MAE), and root mean squared error (RMSE).

Value

An object of class `"approx_convergence"` and `"data.frame"` containing the columns `n`, `max_absolute_error`, `mae`, and `rmse`.

`error_table` *Create an Approximation Error Table*

Description

Creates a numerical error table for an approximation operator. The table may be computed for the operator's current value of `n` or for several values of `n`.

Usage

```
error_table(operator, f, x, n = NULL, digits = 6)
```

Arguments

`operator`	An object of class `"approx_operator"`.
`f`	Function to be approximated.
`x`	Numeric vector of evaluation points.
`n`	Optional vector of positive integers. If `NULL`, the value of `n` stored in `operator` is used.
`digits`	Number of digits used when printing the table.

Value

An object of class `"approx_error_table"` and `"data.frame"` containing pointwise absolute approximation errors.

moments — *Compute Moments of an Approximation Operator*

Description

Computes ordinary or central moments of a positive linear approximation operator.

Usage

```
moments(operator, order = 8L, x, central = FALSE)
```

Arguments

`operator`	An object of class `"approx_operator"`.
`order`	Non-negative integer specifying the maximum moment order. In version 0.1.0, orders from 0 to 8 are supported.
`x`	Numeric vector of evaluation points.
`central`	Logical. If `FALSE`, ordinary moments are returned. If `TRUE`, central moments are returned.

Details

The ordinary moment of order r is m_r(x) = L_n(t^r; x), while the central moment is mu_r(x) = L_n((t - x)^r; x).

If a registered moment engine is available for the operator, it is used to compute the ordinary moments. Otherwise, the ordinary moments are computed through repeated calls to `approximate()`.

Value

A matrix whose rows correspond to evaluation points and whose columns correspond to moment orders from 0 to `order`.

parameter_scan — *Scan an Operator Parameter*

Description

Evaluates the approximation performance of a parametric approximation operator over a sequence of values of one parameter.

Usage

```
parameter_scan(operator, parameter, values, f, grid, n = NULL)
```

Arguments

`operator`	An object of class `"approx_operator"`.
`parameter`	Character string giving the name of the parameter to be varied.
`values`	Numeric vector containing the parameter values to scan.
`f`	Function to be approximated.
`grid`	Numeric vector of evaluation points.
`n`	Optional positive integer giving the operator degree. If `NULL`, the value stored in `operator` is used.

Details

For each parameter value, the operator is reconstructed and the approximation error is evaluated on the supplied grid. The function reports the maximum absolute error, mean absolute error (MAE), and root mean squared error (RMSE).

Value

A data frame of class `c("approx_parameter_scan", "data.frame")` containing the parameter values and the corresponding error measures.

Examples

```
C <- approx_operator(
  family = "sheffer",
  subfamily = "charlier",
  n = 20,
  params = list(a = 2)
)

parameter_scan(
  C,
  parameter = "a",
  values = c(1.5, 2, 3, 5),
  f = function(x) x^2,
  grid = seq(0, 1, length.out = 101)
)
```

`parameter_scan_2d` *Scan Two Operator Parameters*

Description

Evaluates the approximation performance of a parametric approximation operator over all combinations of two parameter sequences.

Usage

```
parameter_scan_2d(
  operator,
  parameter1,
  values1,
  parameter2,
  values2,
  f,
  grid,
  n = NULL
)
```

Arguments

operator	An object of class "approx_operator".
parameter1	Character string giving the name of the first parameter to be varied.
values1	Numeric vector containing the values of the first parameter.
parameter2	Character string giving the name of the second parameter to be varied.
values2	Numeric vector containing the values of the second parameter.
f	Function to be approximated.
grid	Numeric vector of evaluation points.
n	Optional positive integer giving the operator degree. If NULL, the value stored in operator is used.

Details

For each pair of parameter values, the operator is reconstructed and the approximation error is evaluated on the supplied grid. The function reports the maximum absolute error, mean absolute error (MAE), and root mean squared error (RMSE).

Invalid parameter combinations do not interrupt the complete scan. They are retained in the result with valid = FALSE, missing error measures, and an explanatory message.

Value

A data frame of class c("approx_parameter_scan_2d", "data.frame") containing all parameter combinations, validity information, and the corresponding error measures.

Examples

```
M <- approx_operator(
  family = "sheffer",
  subfamily = "meixner",
  n = 20,
  params = list(
    alpha = 0.5,
    beta = 2,
    theta = 1,
```

```
    c = 0.75
  )
)

parameter_scan_2d(
  M,
  parameter1 = "alpha",
  values1 = c(0.5, 1, 2),
  parameter2 = "beta",
  values2 = c(1, 2, 4),
  f = function(x) x^2,
  grid = seq(0, 1, length.out = 101)
)
```

`plot.approx_convergence`

Plot Numerical Convergence

Description

Produces a graphical representation of the numerical convergence measures computed by `convergence()`.

Usage

```
## S3 method for class 'approx_convergence'
plot(
  x,
  metric = c("max_absolute_error", "mae", "rmse"),
  xlab = "n",
  ylab = NULL,
  main = NULL,
  type = "b",
  lwd = NULL,
  pch = NULL,
  show_n = TRUE,
  grid_lines = TRUE,
  ...
)
```

Arguments

`x`	An object of class `"approx_convergence"`.
`metric`	Error measure to be plotted. One of `"max_absolute_error"`, `"mae"`, or `"rmse"`.
`xlab`	Label for the horizontal axis.
`ylab`	Optional label for the vertical axis.
`main`	Optional plot title.

type	Plot type passed to `graphics::plot()`.
lwd	Line width.
pch	Plotting symbol.
show_n	Logical. If `TRUE`, the horizontal axis displays the actual values of n used in the convergence study.
grid_lines	Logical value indicating whether light grid lines should be displayed.
...	Additional graphical arguments passed to `graphics::plot()`.

Details

The horizontal axis represents the parameter n, while the vertical axis represents one of the available error measures: maximum absolute error, mean absolute error (MAE), or root mean squared error (RMSE).

Value

The convergence object `x`, returned invisibly.

`plot.approx_operator` *Plot an Approximation Operator*

Description

Produces graphical representations of the approximation of a function by an approximation operator.

Usage

```
## S3 method for class 'approx_operator'
plot(
  x,
  f,
  grid,
  n = NULL,
  type = c("approximation", "error"),
  xlab = "x",
  ylab = NULL,
  main = NULL,
  lwd = NULL,
  legend_position = "best",
  legend_title = NULL,
  grid_lines = TRUE,
  ...
)
```

Arguments

`x`	An object of class `"approx_operator"`.
`f`	Function to be approximated.
`grid`	Numeric vector of evaluation points.
`n`	Optional vector of positive integers. If `NULL`, the value of `n` stored in the operator is used.
`type`	Type of plot. One of `"approximation"` or `"error"`.
`xlab`	Label for the horizontal axis.
`ylab`	Optional label for the vertical axis.
`main`	Optional plot title.
`lwd`	Line width used for the approximation curves.
`legend_position`	Position of the legend. The default `"best"` places the legend automatically in a low-density region of the graph. The value `"right"` places the legend outside the plotting region. Standard base R legend positions are also supported.
`legend_title`	Optional title for the legend.
`grid_lines`	Logical value indicating whether light grid lines should be displayed.
`...`	Additional graphical arguments passed to `graphics::plot()`.

Details

Two plot types are currently supported:

- `"approximation"` displays the exact function together with the operator approximations for one or more values of n.
- `"error"` displays the absolute approximation error for one or more values of n.

Value

The approximation data used to construct the plot, returned invisibly.

`plot.approx_operator_comparison` *Plot an Approximation Operator Comparison*

Description

Produces a graphical comparison of approximation errors for several approximation operators and values of n.

Usage

```
## S3 method for class 'approx_operator_comparison'
plot(
  x,
  metric = c("max_absolute_error", "mae", "rmse"),
  type = "b",
  lwd = NULL,
  pch = NULL,
  show_n = TRUE,
  show_legend = TRUE,
  legend_position = "best",
  legend_title = "Operator",
  grid_lines = TRUE,
  xlab = "n",
  ylab = NULL,
  main = NULL,
  ...
)
```

Arguments

`x`	An object of class `"approx_operator_comparison"`.
`metric`	Error metric to be plotted. One of `"max_absolute_error"`, `"mae"`, or `"rmse"`.
`type`	Plot type passed to `graphics::lines`.
`lwd`	Line width.
`pch`	Plotting symbol used for the first operator. If `NULL`, the standard ApproxOp plotting symbols are used.
`show_n`	Logical. If `TRUE`, the actual values of `n` are displayed on the horizontal axis.
`show_legend`	Logical. If `TRUE`, a legend is displayed.
`legend_position`	Position of the legend. The default `"best"` places the legend automatically in a low-density region of the graph. The value `"right"` places the legend outside the plotting region. Standard base R legend positions are also supported.
`legend_title`	Optional title for the legend.
`grid_lines`	Logical value indicating whether light grid lines should be displayed.
`xlab`	Label for the horizontal axis.
`ylab`	Optional label for the vertical axis.
`main`	Optional plot title.
`...`	Additional graphical arguments passed to `graphics::plot`.

Details

The available error metrics are the maximum absolute error, mean absolute error (MAE), and root mean squared error (RMSE).

Value

The object x, invisibly.

---

plot.approx_parameter_scan

Plot an Approximation Parameter Scan

Description

Produces a graphical representation of approximation error as a function of a scanned operator parameter.

Usage

```
## S3 method for class 'approx_parameter_scan'
plot(
  x,
  metric = c("max_absolute_error", "mae", "rmse"),
  type = "b",
  lwd = NULL,
  pch = NULL,
  show_values = TRUE,
  grid_lines = TRUE,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  ...
)
```

Arguments

x	An object of class "approx_parameter_scan".
metric	Error metric to be plotted. One of "max_absolute_error", "mae", or "rmse".
type	Plot type used to display the parameter scan.
lwd	Optional line width. If NULL, the ApproxOp graphical default is used.
pch	Optional plotting symbol. If NULL, the ApproxOp graphical default is used.
show_values	Logical. If TRUE, the scanned parameter values are displayed explicitly on the horizontal axis.
grid_lines	Logical. If TRUE, light grid lines are displayed.
xlab	Optional label for the horizontal axis. If NULL, the scanned parameter name is used.
ylab	Optional label for the vertical axis.
main	Optional plot title.
...	Additional graphical arguments passed to graphics::plot().

Details

The available error metrics are the maximum absolute error, mean absolute error (MAE), and root mean squared error (RMSE).

Value

The object x, invisibly.

Examples

```
C <- approx_operator(
  family = "sheffer",
  subfamily = "charlier",
  n = 20,
  params = list(a = 2)
)

Scan <- parameter_scan(
  C,
  parameter = "a",
  values = c(1.5, 2, 3, 5, 10),
  f = function(x) x^2,
  grid = seq(0, 1, length.out = 101)
)

plot(Scan)
plot(Scan, metric = "mae")
```

---

plot.approx_parameter_scan_2d

Plot a Two-Dimensional Parameter Scan

Description

Produces a contour plot for the results obtained from `parameter_scan_2d()`. The plot shows how a selected approximation error metric changes with respect to two operator parameters.

Usage

```
## S3 method for class 'approx_parameter_scan_2d'
plot(
  x,
  metric = c("max_absolute_error", "mae", "rmse"),
  levels = 10,
  draw_points = TRUE,
  show_invalid = TRUE,
  pch = NULL,
```

```
  invalid_pch = 4,
  lwd = NULL,
  grid_lines = TRUE,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  ...
)
```

Arguments

x	An object of class "approx_parameter_scan_2d".
metric	Error metric to be plotted. One of "max_absolute_error", "mae", or "rmse".
levels	Number of contour levels, or a numeric vector specifying the contour levels.
draw_points	Logical. If TRUE, valid parameter combinations are displayed as points.
show_invalid	Logical. If TRUE, invalid parameter combinations are displayed using a different plotting symbol.
pch	Optional plotting symbol for valid parameter combinations. If NULL, the ApproxOp graphical default is used.
invalid_pch	Plotting symbol for invalid parameter combinations.
lwd	Optional contour line width. If NULL, the ApproxOp graphical default is used.
grid_lines	Logical. If TRUE, light grid lines are displayed.
xlab	Optional label for the horizontal axis.
ylab	Optional label for the vertical axis.
main	Optional plot title.
...	Additional graphical arguments passed to graphics::contour().

Details

Invalid parameter combinations are represented by missing values and are therefore excluded from the contour surface.

Value

The object x, invisibly.

print.approx_error_table *Print an Approximation Error Table*

Description

Prints an object of class "approx_error_table" in a structured table with horizontal and vertical separators.

Usage

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

Arguments

x	An object of class "approx_error_table".
...	Additional arguments, currently unused.

Value

The object x, invisibly.

print.approx_operator *Print an Approximation Operator*

Description

Prints a concise summary of an approximation operator object.

Usage

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

Arguments

x	An object of class "approx_operator".
...	Additional arguments, currently unused.

Value

The operator object, invisibly.

print.approx_operator_check *Print an Approximation Operator Check*

Description

Prints a compact summary of the diagnostic results produced by `check_operator()`.

Usage

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

Arguments

`x`	An object of class `"approx_operator_check"`.
`...`	Additional arguments, currently unused.

Value

The object `x`, invisibly.

register_operator *Register a Custom Approximation Operator*

Description

Registers a user-defined approximation operator so that it can be created with `approx_operator()` and used with the computational tools provided by ApproxOp.

Usage

```
register_operator(
  family,
  domain,
  evaluation_engine,
  subfamily = NULL,
  variant = "discrete",
  moment_engine = NULL,
  parameter_validator = NULL,
  display_name = NULL,
  metadata = list(),
  overwrite = FALSE
)
```

Arguments

- `family` Character string specifying the operator family.
- `domain` Numeric vector of length two giving the operator domain.
- `evaluation_engine` Function used to evaluate the operator. It must accept the arguments `f`, `x`, and `operator`, and return a numeric vector of length `length(x)`.
- `subfamily` Optional character string specifying the operator subfamily. The default is `NULL`.
- `variant` Character string specifying the operator variant. The default is `"discrete"`.
- `moment_engine` Optional function for computing ordinary moments. It must accept the arguments `operator`, `order`, and `x`. The default is `NULL`.
- `parameter_validator` Optional function used to validate the parameter list supplied through `approx_operator()`. The function must accept one argument containing the parameter list.
- `display_name` Optional character string used as a descriptive name for the operator.
- `metadata` Optional list containing additional operator metadata.
- `overwrite` Logical. If `TRUE`, an existing registration with the same family, subfamily, and variant may be replaced.

Details

A registered operator must provide an evaluation engine. A moment engine and a parameter validator are optional. When no moment engine is supplied, moments are computed through the general `approximate()` mechanism.

Registration is atomic: if registration fails, the operator and family registries are restored to their previous state.

Value

Invisibly returns `TRUE`.

`verify_moments` *Verify moments of an approximation operator*

Description

Compares moments computed by the moment engine with moments obtained directly from the numerical evaluation of the operator.

Usage

```
verify_moments(operator, order = 8, x, central = FALSE, tolerance = 1e-10)
```

Arguments

`operator`	An object of class `approx_operator`.
`order`	Maximum moment order to verify. Default is 8.
`x`	Evaluation point or vector of evaluation points.
`central`	Logical. If `TRUE`, central moments are verified.
`tolerance`	Numerical tolerance used to determine whether the verification is successful.

Value

A data frame containing the moment order, computed value, numerical value, absolute error, relative error, and verification status.

Index

approx_error, 2
approx_operator, 3
approximate, 4
approximation_data, 4

check_operator, 5
compare_operators, 6
convergence, 6

error_table, 7

moments, 8

parameter_scan, 8
parameter_scan_2d, 9
plot.approx_convergence, 11
plot.approx_operator, 12
plot.approx_operator_comparison, 13
plot.approx_parameter_scan, 15
plot.approx_parameter_scan_2d, 16
print.approx_error_table, 18
print.approx_operator, 18
print.approx_operator_check, 19

register_operator, 19

verify_moments, 20