Package 'speccurvieR'

January 24, 2024

Type Package

Title Easy, Fast, and Pretty Specification Curve Analysis

Version 0.3.0

Description Making specification curve analysis easy, fast, and pretty. It improves upon existing offerings with additional features and 'tidyverse' integration. Users can easily visualize and evaluate how their models behave under different specifications with a high degree of customization. For a description and applications of specification curve analysis see Simonsohn, Simmons, and Nelson (2020) <doi:10.1038/s41562-020-0912-z>.

Encoding UTF-8

LazyData true

Imports ggplot2, magrittr, tidyr, dplyr, stringr, combinat, fixest, pbapply, parallel

RoxygenNote 7.2.3

License MIT + file LICENSE

URL https://github.com/zaynesember/speccurvieR

BugReports https://github.com/zaynesember/speccurvieR/issues

NeedsCompilation no

Author Zayne Sember [aut, cre, cph]

Maintainer Zayne Sember <zsember@ucsd.edu>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2024-01-24 19:40:02 UTC

R topics documented:

`bottles` *CalCOFI Bottle Data*

Description

A subset of data from the California Cooperative Oceanic Fisheries Investigations. Each observation describes a sample of ocean water collected.

Usage

```
bottles
```

Format

'bottles' A data frame with 500 rows and 62 columns:

Cst_Cnt Cast count

Btl_Cnt Bottle Count

Sta_ID Line and Station

Depth_ID Depth ID

Depthm Bottle depth in meters

T_degC Water temperature in degrees Celsius

Salnty Salinity (Practical Salinity Scale 1978)

O2ml_L Milliliters of oxygen per liter of seawater

STheta Potential Density (Sigma Theta), Kg/M³

O2Sat Oxygen percent saturation

Oxy_µmol/Kg Oxygen micromoles per kilogram seawater

BtlNum Niskin bottle sample was collected from

RecInd Record Indicator

T_prec Temperature Precision

T_qual Quality Code

S_prec Salinity Precision

S_qual Quality Code

P_qual Quality Code

O_qual Quality Code

SThtaq Quality Code

O2Satq Quality Code

ChlorA Micrograms Chlorophyll-a per liter seawater

Chlqua Quality Code

Phaeop Micrograms Phaeopigment per liter seawater

Phaqua Quality Code

PO4uM Micromoles Phosphate per liter of seawater

PO4q Quality Code

SiO3uM Micromoles Silicate per liter of seawater

SiO3qu Quality Code

NO2uM Micromoles Nitrite per liter of seawater

NO2q Quality Code

NO3uM Micromoles Nitrate per liter of seawater

NO3q Quality Code

NH3uM Micromoles Ammonia per liter of seawater

NH3q Quality Code

C14As1 14C Assimilation of Replicate 1

C14A1p Precision of 14C Assimilation of Replicate 1

C14A1q Quality Code

C14As2 14C Assimilation of Replicate 2

C14A2p Precision of 14C Assimilation of Replicate 2

C14A2q Quality Code

DarkAs 14C Assimilation of Dark/Control Bottle

DarkAp Precision of 14C Assimilationof Dark/Control Bottle

Darkaq Quality Code

MeanAs Mean 14C Assimilation of Replicates 1 and 2

MeanAp Precision of Mean 14C Assimilation of Replicates 1 and 2

MeanAq Quality Code

IncTim Elapsed incubation time of primary productivity experiment

LightP Light intensities of the incubation tubes

R_Depth Reported Depth (from pressure) in meters

R_Temp Reported (Potential) Temperature in degrees Celsius

R_Sal Reported Salinity (from Specific Volume Anomoly, M³/Kg)

R_DYNHT Reported Dynamic Height in units of dynamic meters

R_Nuts Reported Ammonium concentration

R_Oxy_µmol/Kg Reported Oxygen micromoles/kilogram

DIC1 Dissolved Inorganic Carbon micromoles per kilogram solution

DIC2 Dissolved Inorganic Carbon on a replicate sample

TA1 Total Alkalinity micromoles per kilogram solution

TA2 Total Alkalinity on a replicate sample

pH1 pH (the degree of acidity/alkalinity of a solution)

pH2 pH on a replicate sample

DIC Quality Comment Quality Comment

Source

<https://calcofi.org/data/oceanographic-data/bottle-database/>

`controlExtractor` *Extracts the control variable names and coefficients from an lm model summary.*

Description

Extracts the control variable names and coefficients from a model summary.

Usage

```
controlExtractor(model, x, feols_model = F)
```

Arguments

`model`	A model summary object.
`x`	A string containing the independent variable name.
`feols_model`	An indicator for whether ‘model‘ is a ‘fixest::feols()‘ model. Defaults to ‘FALSE‘.

Value

A dataframe with two columns, ‘term‘ contains the name of the control and ‘coef‘ contains the coefficient estimate.

Examples

```
m <- summary(lm(Salnty ~ STheta + T_degC, bottles))
controlExtractor(model = m, x = "STheta");

m <- summary(lm(Salnty ~ STheta*T_degC + O2Sat, bottles))
controlExtractor(model = m, x = "STheta");
```

duplicate_remover *Removes duplicate control variables*

Description

Removes duplicate control variables from user input.

Usage

```
duplicate_remover(controls, x)
```

Arguments

controls	A vector of strings containing control variable names.
x	A string containing the independent variable name.

Value

A vector of strings containing control variable names

Examples

```
duplicate_remover(controls = c("control1", "control2*control3"),
                  x = "independentVariable");
```

formula_builder *Builds models formulae with every combination of control variables possible.*

Description

Builds models formulae with every combination of control variables possible.

Usage

```
formula_builder(y, x, controls, fixedEffects = NA)
```

Arguments

y	A string containing the dependent variable name.
x	A string containing the independent variable name.
controls	A vector of strings containing control variable names.
fixedEffects	A string containing the name of a variable to use for fixed effects, defaults to 'NA' indicating no fixed effects desired.

Value

A vector of formula objects using every possible combination of controls.

Examples

```
formula_builder("dependentVariable", "independentVariable",
                c("control1", "control2"));
formula_builder("dependentVariable", "independentVariable",
                c("control1*control2"), fixedEffects="month");
```

`paste_factory` *Paste together controls and independent variable*

Description

'paste_factory()' constructs the right hand side of the regression as a a string i.e. "x + control1 + control2".

Usage

```
paste_factory(controls, x)
```

Arguments

`controls`	A vector of strings containing control variable names.
`x`	A string containing the independent variable name.

Value

A string concatenating independent and control variables separated by '+'.

Examples

```
paste_factory(controls = c("control1", "control2"),
              x = "independentVariable");
```

plotAIC *Plots the AIC across model specifications.*

Description

plotAIC() plots the Akaike information criterion across model specifications. Only available for nonlinear regression models.

Usage

```
plotAIC(sca_data, title = "", showIndex = TRUE, plotVars = TRUE)
```

Arguments

sca_data	A data frame returned by ‘sca()‘ containing model estimates from the specification curve analysis.
title	A string to use as the plot title. Defaults to an empty string, ‘""‘.
showIndex	A boolean indicating whether to label the model index on the the x-axis. Defaults to ‘TRUE‘.
plotVars	A boolean indicating whether to include a panel on the plot showing which variables are present in each model. Defaults to ‘TRUE‘.

Value

If ‘plotVars = TRUE‘ returns a grid grob (i.e. the output of a call to ‘grid.draw‘). If ‘plotVars = FALSE‘ returns a ggplot object.

Examples

```
plotAIC(sca_data = sca(y = "Salnty", x = "T_degC",
                       controls = c("ChlorA", "O2Sat"),
                       data = bottles, progressBar = TRUE, parallel = FALSE),
                     title = "AIC");
plotAIC(sca_data = sca(y = "Salnty", x = "T_degC",
                       controls = c("ChlorA*O2Sat"),
                       data = bottles, progressBar = FALSE,
                       parallel = FALSE),
                       showIndex = FALSE, plotVars = FALSE);
plotAIC(sca_data = sca(y = "Salnty", x = "T_degC",
                         controls = c("ChlorA*NO3uM", "O2Sat*NO3uM"),
                         data = bottles,
                         progressBar = TRUE, parallel = TRUE, workers = 2));
```

`plotControlDistributions` *Plots control variable distributions.*

Description

plotControlDistributions() plots the distribution of coefficients for each control variable included in the model specifications.

Usage

```
plotControlDistributions(sca_data, title = "", type = "density")
```

Arguments

`sca_data`	A data frame returned by ‘sca()‘ containing model estimates from the specification curve analysis.
`title`	A string to use as the plot title. Defaults to an empty string, ‘""‘.
`type`	A string indicating what type of distribution plot to produce. When ‘type = "density"‘ density plots are produced. When ‘type = "hist"‘ or ‘type = "histogram"‘ histograms are produced. Defaults to ‘"density"‘.

Value

A ggplot object.

Examples

```
plotControlDistributions(sca_data = sca(y="Salnty", x="T_degC",
                                        controls = c("ChlorA", "O2Sat"),
                                        data = bottles,
                                        progressBar = TRUE, parallel = FALSE),
                         title = "Control Variable Distributions")
plotControlDistributions(sca_data = sca(y = "Salnty", x="T_degC",
                                        controls = c("ChlorA*O2Sat"),
                                        data = bottles,
                                        progressBar = FALSE, parallel = FALSE),
                         type = "hist")
plotControlDistributions(sca_data = sca(y = "Salnty", x = "T_degC",
                                        controls = c("ChlorA*NO3uM",
                                                     "O2Sat*NO3uM"),
                                        data = bottles, progressBar = TRUE,
                                        parallel = TRUE, workers = 2),
                         type = "density")
```

plotCurve *Plots a specification curve.*

Description

plotCurve() takes the data frame output of sca() and produces a ggplot of the independent variable's coefficient (as indicated in the call to sca()) across model specifications. By default a panel is added showing which control variables are present in each model. Note that the ggplot output by this function can only be further customized when 'plotVars = FALSE', i.e. when the control variable panel is not included.

Usage

```
plotCurve(
  sca_data,
  title = "",
  showIndex = TRUE,
  plotVars = TRUE,
  ylab = "Coefficient",
  plotSE = "bar"
)
```

Arguments

sca_data	A data frame returned by 'sca()' containing model estimates from the specification curve analysis.
title	A string to use as the plot title. Defaults to an empty string, '""'.
showIndex	A boolean indicating whether to label the model index on the the x-axis. Defaults to 'TRUE'.
plotVars	A boolean indicating whether to include a panel on the plot showing which variables are present in each model. Defaults to 'TRUE'.
ylab	A string to be used as the y-axis label. Defaults to '"Coefficient"'.
plotSE	A string indicating whether to display standard errors as bars or plots. For bars 'plotSE = "bar"', for ribbons 'plotSE = "ribbon"'. If any other value is supplied then no standard errors are included. Defaults to '"bar"'.

Value

If 'plotVars = TRUE' returns a grid grob (i.e. the output of a call to 'grid.draw'). If 'plotVars = FALSE' returns a ggplot object.

Examples

```
plotCurve(sca_data = sca(y="Salnty", x="T_degC", c("ChlorA", "O2Sat"),
                         data=bottles, progressBar=TRUE, parallel=FALSE),
                      title = "Salinity and Temperature Models",
```

```
                      showIndex = TRUE, plotVars = TRUE,
                      ylab = "Coefficient value", plotSE = "ribbon");
plotCurve(sca_data = sca(y="Salnty", x="T_degC",
                         c("ChlorA*O2Sat", "ChlorA", "O2Sat"),
                         data=bottles, progressBar=FALSE, parallel=FALSE),
                      showIndex = TRUE, plotVars = TRUE,
                      plotSE = "ribbon");
plotCurve(sca_data = sca(y="Salnty", x="T_degC",
                         c("ChlorA*NO3uM", "O2Sat", "ChlorA", "NO3uM"),
                         data=bottles,
                         progressBar = TRUE, parallel = TRUE, workers=2),
          plotSE="");
```

`plotDeviance` *Plots the deviance of residuals across model specifications.*

Description

plotDeviance() plots the deviance of residuals across model specifications. Only available for linear regression models.

Usage

```
plotDeviance(sca_data, title = "", showIndex = TRUE, plotVars = TRUE)
```

Arguments

`sca_data`	A data frame returned by 'sca()' containing model estimates from the specification curve analysis.
`title`	A string to use as the plot title. Defaults to an empty string, '""'.
`showIndex`	A boolean indicating whether to label the model index on the the x-axis. Defaults to 'TRUE'.
`plotVars`	A boolean indicating whether to include a panel on the plot showing which variables are present in each model. Defaults to 'TRUE'.

Value

If 'plotVars = TRUE' returns a grid grob (i.e. the output of a call to 'grid.draw'). If 'plotVars = FALSE' returns a ggplot object.

Examples

```
plotDeviance(sca_data = sca(y = "Salnty", x = "T_degC",
                            controls = c("ChlorA", "O2Sat"),
                            data = bottles, progressBar = TRUE,
                            parallel = FALSE),
                      title = "Model Deviance");
plotDeviance(sca_data = sca(y = "Salnty", x = "T_degC",
```

```
                            controls = c("ChlorA*O2Sat"),
                            data = bottles, progressBar = FALSE,
                            parallel = FALSE),
                        showIndex = FALSE, plotVars = FALSE);
plotDeviance(sca_data = sca(y = "Salnty", x="T_degC",
                            controls = c("ChlorA*NO3uM", "O2Sat*NO3uM"),
                            data = bottles, progressBar = TRUE, parallel = TRUE,
                            workers = 2));
```

---

plotR2Adj *Plots the adj. R-squared across model specifications.*

Description

plotR2Adj() plots the adjusted R-squared across model specifications. Only available for linear regression models. Note when fixed effects are are specified the within adjusted R-squared is used (i.e. ‘fixest::r2()‘ with ‘type="war2"‘).

Usage

```
plotR2Adj(sca_data, title = "", showIndex = TRUE, plotVars = TRUE)
```

Arguments

sca_data	A data frame returned by ‘sca()‘ containing model estimates from the specification curve analysis.
title	A string to use as the plot title. Defaults to an empty string, ‘""‘.
showIndex	A boolean indicating whether to label the model index on the the x-axis. Defaults to ‘TRUE‘.
plotVars	A boolean indicating whether to include a panel on the plot showing which variables are present in each model. Defaults to ‘TRUE‘.

Value

If ‘plotVars = TRUE‘ returns a grid grob (i.e. the output of a call to ‘grid.draw‘). If ‘plotVars = FALSE‘ returns a ggplot object.

Examples

```
plotR2Adj(sca_data = sca(y = "Salnty", x = "T_degC",
                            controls = c("ChlorA", "O2Sat"),
                            data = bottles, progressBar = TRUE,
                            parallel = FALSE),
                        title = "Adjusted R^2");
plotR2Adj(sca_data = sca(y="Salnty", x="T_degC",
                            controls = c("ChlorA*O2Sat"),
                            data = bottles, progressBar = FALSE,
                            parallel = FALSE),
```

```
                      showIndex = FALSE, plotVars = FALSE);
plotR2Adj(sca_data = sca(y = "Salnty", x = "T_degC",
                         controls = c("ChlorA*NO3uM", "O2Sat*NO3uM"),
                         data = bottles,
                         progressBar = TRUE, parallel = TRUE, workers = 2));
```

---

plotRMSE *Plots RMSE across model specifications.*

Description

plotRMSE() plots the root mean square error across model specifications. Only available for linear regression models.

Usage

```
plotRMSE(sca_data, title = "", showIndex = TRUE, plotVars = TRUE)
```

Arguments

sca_data	A data frame returned by 'sca()' containing model estimates from the specification curve analysis.
title	A string to use as the plot title. Defaults to an empty string, '""'.
showIndex	A boolean indicating whether to label the model index on the the x-axis. Defaults to 'TRUE'.
plotVars	A boolean indicating whether to include a panel on the plot showing which variables are present in each model. Defaults to 'TRUE'.

Value

If 'plotVars = TRUE' returns a grid grob (i.e. the output of a call to 'grid.draw'). If 'plotVars = FALSE' returns a ggplot object.

Examples

```
plotRMSE(sca_data = sca(y="Salnty", x="T_degC", c("ChlorA", "O2Sat"),
                        data=bottles, progressBar=TRUE, parallel=FALSE),
                      title = "RMSE");
plotRMSE(sca_data = sca(y="Salnty", x="T_degC", c("ChlorA*O2Sat"),
                        data=bottles, progressBar=FALSE, parallel=FALSE),
                      showIndex = FALSE, plotVars = FALSE);
plotRMSE(sca_data = sca(y="Salnty", x="T_degC",
                        c("ChlorA*NO3uM", "O2Sat*NO3uM"), data=bottles,
                        progressBar = TRUE, parallel=TRUE, workers=2));
```

plotVars *Plots the variables in each model.*

Description

plotVars() plots the variables included in each model specification in order of model index. Returns a ggplot object that can then be combined with the output of other functions like plotRMSE() if further customization of each plot is desired.

Usage

```
plotVars(sca_data, title = "", colorControls = FALSE)
```

Arguments

sca_data	A data frame returned by 'sca()' containing model estimates from the specification curve analysis.
title	A string to use as the plot title. Defaults to an empty string, '""'.
colorControls	A boolean indicating whether to give each variable a color to improve readability. Defaults to 'FALSE'.

Value

A ggplot object.

Examples

```
plotVars(sca_data = sca(y = "Salnty", x = "T_degC",
                        controls = c("ChlorA", "O2Sat"),
                        data = bottles, progressBar = TRUE,
                        parallel = FALSE),
                      title = "Model Variable Specifications");
plotVars(sca_data = sca(y = "Salnty", x = "T_degC",
                        controls = c("ChlorA*O2Sat"),
                        data = bottles, progressBar = FALSE,
                        parallel = FALSE),
                     colorControls = TRUE);
plotVars(sca_data = sca(y = "Salnty", x = "T_degC",
                        controls = c("ChlorA*NO3uM", "O2Sat*NO3uM"),
                        data = bottles,
                        progressBar = TRUE, parallel = TRUE, workers = 2));
```

sca *Perform specification curve analysis*

Description

sca() is the workhorse function of the package–this estimates models with every possible combination of the controls supplied and returns a data frame where each row contains the pertinent information and parameters for a given model by default. This data frame can then be input to plotCurve() or any other plotting function in the package. Alternatively, if 'returnFormulae = TRUE', it returns a list of formula objects with every possible combination of controls.

Usage

```
sca(
  y,
  x,
  controls,
  data,
  family = "linear",
  link = NULL,
  fixedEffects = NULL,
  returnFormulae = FALSE,
  progressBar = TRUE,
  parallel = FALSE,
  workers = 2
)
```

Arguments

y	A string containing the column name of the dependent variable in data.
x	A string containing the column name of the independent variable in data.
controls	A vector of strings containing the column names of the control variables in data.
data	A dataframe containing y, x, controls, and (optionally) the variables to be used for fixed effects or clustering.
family	A string indicating the family of models to be used. Defaults to "linear" for OLS regression but supports all families supported by 'glm()'.
link	A string specifying the link function to be used for the model. Defaults to 'NULL' for OLS regression using 'lm()' or 'fixest::feols()' depending on whether fixed effects are supplied. Supports all link functions supported by the family parameter of 'glm()'.
fixedEffects	A string containing the column name of the variable in data desired for fixed effects. Defaults to NULL in which case no fixed effects are included.
returnFormulae	A boolean. When 'TRUE' a list of model formula objects is returned but the models are not estimated. Defaults to 'FALSE' in which case a dataframe of model results is returned.

progressBar	A boolean indicating whether the user wants a progress bar for model estimation. Defaults to 'TRUE'.
parallel	A boolean indicating whether to parallelize model estimation. Parallelization only offers a speed advantage when a large (> 1000) number of models is being estimated. Defaults to 'FALSE'.
workers	An integer indicating the number of workers to use for parallelization. Defaults to 2.

Value

When 'returnFormulae' is 'FALSE', a dataframe where each row contains the independent variable coefficient estimate, standard error, test statistic, p-value, model specification, and measures of model fit.

Examples

```
sca(y = "Salnty", x = "T_degC", controls = c("ChlorA", "O2Sat"),
    data = bottles, progressBar = TRUE, parallel = FALSE);
sca(y = "Salnty", x = "T_degC", controls = c("ChlorA*NO3uM", "O2Sat*NO3uM"),
    data = bottles, progressBar = TRUE, parallel = TRUE, workers = 2);
sca(y = "Salnty", x = "T_degC", controls = c("ChlorA", "O2Sat*NO3uM"),
    data = bottles, progressBar = TRUE, parallel = FALSE,
    returnFormulae = TRUE);
```

---

scp *Prepares the output of 'sca()' for plotting.*

Description

Takes in the data frame output by 'sca()' and returns a list with the data frame and labels to make a plot to visualize the controls included in each spec curve model.

Usage

```
scp(sca_data)
```

Arguments

sca_data	A data frame output by 'sca'.

Value

A list containing a data frame, control coefficients, and control names.

Examples

```
scp(sca(y = "Salnty", x = "T_degC", controls = c("ChlorA", "O2Sat"),
        data = bottles, progressBar=TRUE, parallel=FALSE));
```

unAsIs *Removes the 'AsIs' class attribute from the input.*

Description

Removes the 'AsIs' class attribute from the input. Taken from: <https://stackoverflow.com/a/12866609>

Usage

```
unAsIs(x)
```

Arguments

x	An object with the 'AsIs' class attribute.

Value

An object without the 'AsIs' class attribute.

Examples

```
unAsIs(x = I(c(1:4)));
```

Index