

Package: kvr2 (via r-universe)

September 22, 2026

Type Package

Title Calculate and Compare Multiple Definitions of Coefficient of Determination

Version 0.2.0.9000

Description Calculate nine types of coefficients of determination (R-squared) based on the classification by Kvalseth (1985) [doi:10.1080/00031305.1985.10479448](https://doi.org/10.1080/00031305.1985.10479448). This package is designed for educational purposes to demonstrate how R-squared values can fluctuate depending on the choice of formula, particularly in power regression models or linear models without an intercept. By providing a comprehensive list of definitions, it helps users understand the mathematical sensitivity of goodness-of-fit indices.

URL <https://github.com/indenkun/kvr2>, <https://indenkun.github.io/kvr2/>

BugReports <https://github.com/indenkun/kvr2/issues>

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Roxygen list(markdown = TRUE)

Imports insight, ggplot2, grid, stats, tidyr

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/pak/sysreqs libicu-dev

Repository <https://indenkun.r-universe.dev>

Date/Publication 2026-03-11 01:15:13 UTC

RemoteUrl <https://github.com/indenkun/kvr2>

RemoteRef HEAD

RemoteSha 5df8d5dc85d67bdf4c11b3709e3e92c555a0b9c1

Contents

comp_fit	2
comp_model	4
model_info	5
plot.comp_model	6
plot.r2_kvr2	7
plot_diagnostic	8
plot_kvr2	8
print.comp_model	10
print.r2_kvr2	11
r2	12
r2_adjusted	15
Index	17

comp_fit	<i>Calculate Comparative Fit Measures for Regression Models</i>
----------	---

Description

Calculates goodness-of-fit metrics based on Kvalseth (1985), including Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Squared Error (MSE). This function provides a unified output for comparing different model specifications.

Usage

```
comp_fit(model, type = c("auto", "linear", "power"))

RMSE(model, type = c("auto", "linear", "power"))

MAE(model, type = c("auto", "linear", "power"))

MSE(model, type = c("auto", "linear", "power"))
```

Arguments

model	A linear model or power regression model of the lm.
type	Character string. Selects the model type: "linear", "power", or "auto" (default). In "auto", the function detects if the dependent variable is log-transformed.

Details

The metrics are calculated according to the formulas in Kvalseth (1985):

- **RMSE:** Root Mean Squared Residual or Error

$$RMSE = \sqrt{\frac{\sum (y - \hat{y})^2}{n}}$$

- **MAE:** Mean Absolute Residual or Error

$$MAE = \frac{\sum |y - \hat{y}|}{n}$$

- **MSE:** Mean Squared Residual or Error (Adjusted for degrees of freedom)

$$MSE = \frac{\sum (y - \hat{y})^2}{n - p}$$

where n is the sample size and p is the number of model parameters (including the intercept).

Note on MSE: In many modern contexts, "MSE" refers to the mean squared error without degree-of-freedom adjustment (denominator n). However, this function follows Kvalseth's definition, which uses $n - p$ as the denominator.

Value

- For `comp_fit()`: An object of class `comp_kvr2`, which is a list containing the calculated RMSE, MAE, and MSE values.
- For individual functions (`RMSE()`, `MAE()`, `MSE()`): A named numeric value of the specific metric.

Note

The power regression model must be based on a logarithmic transformation.

When `type = "auto"`, the choice between linear and power regression is determined by analyzing the model formula. It identifies a power regression if the dependent variable is a function call to `log()` (e.g., `lm(log(y) ~ x)`).

Note that simple variable names containing the string "log" (e.g., `lm(log_value ~ x)`) are correctly treated as linear regression. To override this automatic detection, manually specify `type = "linear"` or `type = "power"`.

References

Tarald O. Kvalseth (1985) Cautionary Note about R^2 , The American Statistician, 39:4, 279-285, doi: [10.1080/00031305.1985.10479448](https://doi.org/10.1080/00031305.1985.10479448)

See Also

`print.comp_kvr2()`

Examples

```
# example data set 1. Kvålseth (1985).
df1 <- data.frame(x = c(1:6),
                  y = c(15,37,52,59,83,92))
model_intercept <- lm(y ~ x, df1)
model_without <- lm(y ~ x - 1, df1)
model_power <- lm(log(y) ~ log(x), df1)
comp_fit(model_intercept)
```

```
comp_fit(model_without)
comp_fit(model_power)
```

comp_model

Contrast R-squared Definitions: Intercept vs. No-Intercept

Description

A specialized tool for educational and diagnostic purposes. This function automatically generates a comparison between a model with an intercept and its forced no-intercept counterpart (or vice versa), revealing how mathematical definitions of R-squared diverge under different constraints.

Usage

```
comp_model(model, type = c("auto", "linear", "power"), adjusted = FALSE)
```

Arguments

model	A linear model or power regression model of the <code>lm</code> .
type	Character string. Selects the model type: "linear", "power", or "auto" (default). In "auto", the function detects if the dependent variable is log-transformed.
adjusted	Logical. If TRUE, calculates the adjusted coefficient of determination for each formula.

Details

This function reconstructs the alternative model using QR decomposition rather than `update()` to ensure robustness against environment/scoping issues.

It is particularly useful for observing how definitions like R_2^2 can exceed 1.0 or R_1^2 can become negative when an intercept is removed, illustrating the "pitfalls" discussed in Kvalseth (1985).

Value

A data frame of class `comp_model` containing nine R-squared definitions and three fit metrics (RMSE, MAE, MSE) for both intercept and no-intercept versions.

The original model objects are stored as attributes `with_int` and `without_int` for use by the plot method.

References

Kvalseth, T. O. (1985) Cautionary Note about R^2 . The American Statistician, 39(4), 279-285.

Examples

```
df1 <- data.frame(x = 1:6, y = c(15, 37, 52, 59, 83, 92))
model <- lm(y ~ x, data = df1)

# Compare R-squared sensitivity
comp_model(model)

# Compare adjusted R-squared
comp_model(model, adjusted = TRUE)
```

model_info

*Get Model Information Used for Calculations***Description**

Extracts the metadata and model specifications used to calculate the coefficients of determination, such as the regression type, sample size, and degrees of freedom.

Usage

```
model_info(x)
```

Arguments

x An object of class `r2_kvr2` or `comp_kvr2`.

Details

This function provides transparency into the calculation process of the various R-squared definitions. It is particularly useful for verifying whether a model was treated as a "power" regression (log-transformed) and how the degrees of freedom were determined for adjusted R-squared values.

Value

A list containing the following components:

- **type**: A string indicating the regression type ("linear" or "power").
- **has_intercept**: A logical value indicating if the model includes an intercept.
- **n**: The number of observations used in the model (excluding missing values).
- **k**: The number of estimated parameters (including the intercept if present).
- **df_res**: Residual degrees of freedom ($n - k$).

Note

The sample size `n` refers to the actual number of observations used by `lm()`, which may be fewer than the rows in the original data frame if NA values were present.

See Also

[r2\(\)](#), [comp_fit\(\)](#)

Examples

```
df1 <- data.frame(x = 1:6, y = c(15, 37, 52, 59, 83, 92))
model <- lm(y ~ x, data = df1)
res <- r2(model)

# Check the metadata
info <- model_info(res)
info$n
info$type
```

plot.comp_model

Plot Comparison of Model Specifications

Description

Generates a comprehensive 2x2 diagnostic dashboard comparing models with and without an intercept. This visualization helps identify how the absence of an intercept affects different R-squared definitions and error metrics.

Usage

```
## S3 method for class 'comp_model'
plot(x, ...)
```

Arguments

x An object of class `comp_model` generated by [comp_model\(\)](#).

... Further graphical parameters (currently ignored).

Details

The plot is organized into four panels:

- **Top-Left:** Grouped bar chart of the nine R-squared definitions.
- **Bottom-Left:** Comparison of absolute fit metrics (RMSE, MAE, MSE).
- **Top-Right:** Observed vs. Predicted plot for the intercept model.
- **Bottom-Right:** Observed vs. Predicted plot for the no-intercept model.

This layout allows for a direct "cause-and-effect" analysis: for instance, observing a data point far from the identity line in the bottom-right panel explains why certain R-squared definitions might crash or become negative in the left panels.

Value

This function is primarily called for its side effect of creating a grid-based plot. It returns the input object `x` invisibly.

Note

Since this plot uses the grid system to combine multiple ggplot objects, it cannot be further modified with the `+` operator. If you need to customize individual panels, use the internal plotting functions or extract the models from the `comp_model` object attributes.

See Also

`comp_model()`, `plot_diagnostic()`

Examples

```
df <- data.frame(x = 1:5, y = c(2, 3, 5, 4, 6))
m1 <- lm(y ~ x, data = df)
res <- comp_model(m1)
plot(res)
```

plot.r2_kvr2

Plot Method for r2_kvr2 Objects

Description

Visualizes the nine definitions of R-squared to compare their values and identify potential issues (e.g., values exceeding 1 or falling below 0).

Usage

```
## S3 method for class 'r2_kvr2'
plot(x, ...)
```

Arguments

`x` An object of class `r2_kvr2`.
`...` Currently ignored.

Value

A ggplot object representing the visual analysis.

Examples

```
df1 <- data.frame(x = 1:6, y = c(15, 37, 52, 59, 83, 92))
model <- lm(y ~ x - 1, data = df1) # No-intercept model
r2(model)
```

plot_diagnostic	<i>Plot Observed vs Predicted Values</i>
-----------------	--

Description

A diagnostic plot to visualize why R-squared might be low or negative. It compares the model predictions (identity line) against the mean (horizontal line).

Usage

```
plot_diagnostic(x, ...)
```

Arguments

x	A fitted lm object.
...	Currently ignored.

Value

A ggplot object representing the visual analysis.

Examples

```
df1 <- data.frame(x = 1:6, y = c(15, 37, 52, 59, 83, 92))
model <- lm(y ~ x - 1, data = df1) # No-intercept model
plot_diagnostic(model)
```

plot_kvr2	<i>Plot Method for Kvalseth's R-squared Objects</i>
-----------	---

Description

Visualizes the different R-squared definitions or provides a diagnostic observed-vs-predicted plot to understand the model fit.

Usage

```
plot_kvr2(
  x,
  type = c("auto", "linear", "power"),
  plot_type = c("both", "r2", "diag"),
  ...
)
```

Arguments

x	An object of class <code>lm</code> .
type	Character string. Selects the model type: "linear", "power", or "auto" (default). In "auto", the function detects if the dependent variable is log-transformed.
plot_type	A string specifying the plot layout: "both" (default) displays the bar plot and diagnostic plot side-by-side, "r2" shows only the R-squared comparison, and "diag" shows only the observed-vs-predicted plot.
...	Currently ignored.

Details

When `plot_type = "r2"`, the function creates a bar plot comparing all nine definitions. Bars are colored based on their validity:

- **Skyblue:** Standard values between 0 and 1.
- **Orange:** Values exceeding 1.0 or falling below 0.0 (warnings).

When `plot_type = "diag"`, the function displays a scatter plot of observed vs. predicted values. Two reference lines are added:

- **Darkgreen Solid Line:** The 1:1 "perfect fit" line (RSS reference).
- **Red Dashed Line:** The overall mean of the observed data (TSS reference).

If the data points are closer to the red dashed line than the green solid line, R_1^2 will be negative.

Combined View (`plot_type = "both"`): Automatically configures the plotting device to show both plots simultaneously for a comprehensive model evaluation.

Value

The return value depends on the `plot_type` argument:

- For "r2" and "diag": Returns a ggplot object that can be further customized.
- For "both": Generates a combined plot using the grid system and returns the input object `x` invisibly.

Examples

```
df1 <- data.frame(x = 1:6, y = c(15, 37, 52, 59, 83, 92))
model <- lm(y ~ x - 1, data = df1) # No-intercept model
plot_kvr2(model)
# Compare all definitions
plot_kvr2(model, plot_type = "r2")

# Diagnostic plot to see why some R2 might be problematic
plot_kvr2(model, plot_type = "diag")
```

print.comp_model	<i>Print Method for Model Comparison Objects</i>
------------------	--

Description

A specialized print method for `comp_model` objects. It formats the comparison table for better readability and provides diagnostic warnings if any R-squared values fall outside the standard 0 to 1 range.

Usage

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

Arguments

<code>x</code>	An object of class <code>comp_model</code> .
<code>...</code>	Further arguments passed to or from other methods.
<code>digits</code>	Number of decimal places to be used for formatting numerical values. Default is 4.

Details

The output is formatted using the `insight` package's `export_table()` functionality, ensuring a clean and structured display in the console.

In addition to the table, this method performs an automated check on the R-squared values (columns 2 to 10). If any value exceeds 1.0 or falls below 0.0, a warning message is displayed. This is a critical educational feature, as it flags instances where specific R^2 definitions become mathematically inappropriate due to the lack of an intercept or model misspecification.

Value

Returns the input object `x` invisibly.

See Also

[comp_model\(\)](#)

print.r2_kvr2	<i>Print Methods for r2 and comp_fit calculation Objects</i>
---------------	--

Description

Printing objects of class "r2_kvr2" (generated by `r2()`) or "comp_kvr2" (generated by `comp_fit()`), respectively, by simple print methods.

Usage

```
## S3 method for class 'r2_kvr2'
print(x, ..., digits = 4, model_info = TRUE)

## S3 method for class 'comp_kvr2'
print(x, ..., digits = 4, model_info = TRUE)
```

Arguments

x	An object of class "r2_kvr2" or "comp_kvr2".
...	Further arguments passed to or from other methods.
digits	The number of decimal places to be used for rounding the results. Default is 4.
model_info	Logical. If TRUE (default), additional information about the model (type, intercept, n, k) is printed below the results.

Details

These methods format the calculated statistics into a human-readable summary, displaying each index or metric with its corresponding value.

Value

The input object is returned invisibly (via `invisible(x)`). This function is called for its side effect of printing the results of `r2()` or `comp_fit()` calculations to the console.

See Also

`r2()` `comp_fit()` `r2_adjusted()`

r2	<i>Calculate Multiple Definitions of Coefficient of Determination (R^2-squared)</i>
----	--

Description

Calculates nine types of coefficients of determination (R^2) based on the classification by Kvalseth (1985). This function is designed to demonstrate how R^2 values can vary depending on their mathematical definition, particularly in models without an intercept or in power regression models

Usage

```
r2(model, type = c("auto", "linear", "power"), adjusted = FALSE)

r2_1(model, type = c("auto", "linear", "power"))
r2_2(model, type = c("auto", "linear", "power"))
r2_3(model, type = c("auto", "linear", "power"))
r2_4(model, type = c("auto", "linear", "power"))
r2_5(model, type = c("auto", "linear", "power"))
r2_6(model, type = c("auto", "linear", "power"))
r2_7(model, type = c("auto", "linear", "power"))
r2_8(model, type = c("auto", "linear", "power"))
r2_9(model, type = c("auto", "linear", "power"))
```

Arguments

model	A linear model or power regression model of the <code>lm</code> .
type	Character string. Selects the model type: "linear", "power", or "auto" (default). In "auto", the function detects if the dependent variable is log-transformed.
adjusted	Logical. If TRUE, calculates the adjusted coefficient of determination for each formula.

Details

The nine coefficient equations from R_1^2 to R_9^2 are based on Kvalseth (1985) and are as follows:

- R_1^2 : Proportion of total variance explained.

$$R_1^2 = 1 - \frac{\sum (y - \hat{y})^2}{\sum (y - \bar{y})^2}$$

- R_2^2 : Based on the variation of predicted values.

$$R_2^2 = \frac{\sum(\hat{y} - \bar{y})^2}{\sum(y - \bar{y})^2}$$

- R_3^2 : Ratio of variation using the mean of predicted values.

$$R_3^2 = \frac{\sum(\hat{y} - \bar{\hat{y}})^2}{\sum(y - \bar{y})^2}$$

- R_4^2 : Incorporates the mean residual.

$$R_4^2 = 1 - \frac{\sum(e - \bar{e})^2}{\sum(y - \bar{y})^2}, \quad e = y - \hat{y}$$

- R_5^2 : The square of the multiple correlation coefficient between the dependent variable and the independent variable (a comprehensive indicator in linearized models).

R_5^2 = squared multiple correlation coefficient between the regressand and the regressors

- R_6^2 : Square of Pearson's correlation coefficient between observed y and predicted $\hat{y}$.

$$R_6^2 = \frac{(\sum(y - \bar{y})(\hat{y} - \bar{\hat{y}}))^2}{\sum(y - \bar{y})^2 \sum(\hat{y} - \bar{\hat{y}})^2}$$

- R_7^2 : Recommended for models without an intercept.

$$R_7^2 = 1 - \frac{\sum(y - \hat{y})^2}{\sum y^2}$$

- R_8^2 : Alternative form for models without an intercept.

$$R_8^2 = \frac{\sum \hat{y}^2}{\sum y^2}$$

- R_9^2 : Robust version using medians to resist outliers.

$$R_9^2 = 1 - \left(\frac{M\{|y_i - \hat{y}_i|\}}{M\{|y_i - \bar{y}|\}} \right)^2$$

where M represents the median of the sample.

For degree of freedom adjustment `adjusted = TRUE`, refer to [r2_adjusted](#).

Value

- For `r2()`: An object of class `r2_kvr2`, which is a list containing calculated values for all R^2 formulas.
- For individual functions (`r2_1()` to `r2_9()`): A named numeric value of the specific R^2 definition. calculated values for each R^2 formula.

Note

The power regression model must be based on a logarithmic transformation.

When `type = "auto"`, the choice between linear and power regression is determined by analyzing the model formula. It identifies a power regression if the dependent variable is a function call to `log()` (e.g., `lm(log(y) ~ x)`).

Note that simple variable names containing the string "log" (e.g., `lm(log_value ~ x)`) are correctly treated as linear regression. To override this automatic detection, manually specify `type = "linear"` or `type = "power"`.

References

Tarald O. Kvalseth (1985) Cautionary Note about R^2 , The American Statistician, 39:4, 279-285, [doi:10.1080/00031305.1985.10479448](https://doi.org/10.1080/00031305.1985.10479448)

Box, George E. P., Hunter, William G., Hunter, J. Stuart. (1978) Statistics for experimenters: an introduction to design, data analysis, and model building. New York, United States, J. Wiley, p. 462-473, ISBN:9780471093152.

See Also

`print.r2_kvr2()` `r2_adjusted()`

Examples

```
# Example data set 1. Kvalseth (1985).
df1 <- data.frame(x = c(1:6),
                  y = c(15,37,52,59,83,92))
# Linear regression model with intercept
model_intercept1 <- lm(y ~ x, df1)
# Linear regression model without intercept
model_without1 <- lm(y ~ x - 1, df1)
# Power regression model
model_power1 <- lm(log(y) ~ log(x), df1)
r2(model_intercept1)
r2(model_without1)
r2(model_power1)
# Example data set 2. Kvalseth (1985).
df2 <- data.frame(x = 6:13,
                  y = c(3882, 1266, 733, 450, 410, 305, 185, 112))
power_model2 <- lm(log((y/7343)) ~ log(x), data = df2)
r2(power_model2)
# Example of a Multiple Regression Analysis Model.
# The data for two independent variables given by Box et al. (1978, p. 462)
# as used in Kvalseth (1985).
df3 <- data.frame(x1 = c(0.34, 0.34, 0.58, 1.26, 1.26, 1.82),
                  x2 = c(0.73, 0.73, 0.69, 0.97, 0.97, 0.46),
                  y = c(5.75, 4.79, 5.44, 9.09, 8.59, 5.09))
# Multiple regression analysis model with intercept
model_intercept3 <- lm(y ~ x1 + x2, df3)
# Multiple regression analysis model without intercept
model_without3 <- lm(y ~ x1 + x2 - 1, df3)
```

```
# Multiple power regression analysis model
model_power3 <- lm(log(y) ~ log(x1) + log(x2), df3)
r2(model_intercept3)
r2(model_without3)
r2(model_power3)
```

r2_adjusted

*Calculate the Adjusted Determination Coefficient***Description**

Calculate the adjusted coefficient of determination by entering the regression model and coefficient of determination. See details.

Usage

```
r2_adjusted(model, r2)
```

Arguments

`model` A linear model or power regression model of the `lm`.
`r2` A numeric. Coefficient of determination.

Details

The adjustment factor a is calculated using the following formula.

$$a = (n - 1)/(n - k - 1)$$

n is the sample size, and k is the number of parameters in the regression model.

R_a^2 ($R^2_{adjusted}$) is calculated using the following formula.

$$R_a^2 = 1 - a(1 - R^2)$$

This function performs freedom-of-degrees adjustment for all coefficients based on the above formula. However, Kvalseth (1985) recommends applying freedom-of-degrees adjustment only to R_1^2 and R_9^2 , based on the principle of consistency in coefficients. Furthermore, there is no basis for applying the same type of adjustment to R_6^2 (the square of the correlation coefficient) or to R_7^2 and R_8^2 , which depend on specific model forms.

For details on each coefficient of determination, refer to [r2\(\)](#).

Value

A numeric vector or a list of class `r2_kvr2` containing the adjusted R^2 values. Each element represents the adjusted version of the corresponding R^2 definition, accounting for the degrees of freedom.

References

Tarald O. Kvalseth (1985) Cautionary Note about R^2 , The American Statistician, 39:4, 279-285,
[doi:10.1080/00031305.1985.10479448](https://doi.org/10.1080/00031305.1985.10479448)

See Also

[r2\(\)](#)

Index

`comp_fit`, [2](#)
`comp_fit()`, [6](#), [11](#)
`comp_model`, [4](#)
`comp_model()`, [6](#), [7](#), [10](#)

`lm()`, [5](#)

`MAE (comp_fit)`, [2](#)
`model_info`, [5](#)
`MSE (comp_fit)`, [2](#)

`plot.comp_model`, [6](#)
`plot.r2_kvr2`, [7](#)
`plot_diagnostic`, [8](#)
`plot_diagnostic()`, [7](#)
`plot_kvr2`, [8](#)
`print.comp_kvr2 (print.r2_kvr2)`, [11](#)
`print.comp_kvr2()`, [3](#)
`print.comp_model`, [10](#)
`print.r2_kvr2`, [11](#)
`print.r2_kvr2()`, [14](#)

`r2`, [12](#)
`r2()`, [6](#), [11](#), [15](#), [16](#)
`r2_1 (r2)`, [12](#)
`r2_2 (r2)`, [12](#)
`r2_3 (r2)`, [12](#)
`r2_4 (r2)`, [12](#)
`r2_5 (r2)`, [12](#)
`r2_6 (r2)`, [12](#)
`r2_7 (r2)`, [12](#)
`r2_8 (r2)`, [12](#)
`r2_9 (r2)`, [12](#)
`r2_adjusted`, [13](#), [15](#)
`r2_adjusted()`, [11](#), [14](#)
`RMSE (comp_fit)`, [2](#)