Package ‘flex’

September 2, 2025

Title Fuzzy Linear Squares Estimation with Explicit Formula (FLEX)

Version 0.1.0

Date 2025-08-27

Description The FLEX method, developed by Yoon and Choi (2013) <doi:10.1007/978-3-642-33042-1_21>, performs least squares estimation for fuzzy predictors and outcomes, generating crisp regression coefficients by minimizing the distance between observed and predicted outcomes. It also provides functions for fuzzifying data and inference tasks, including significance testing, fit indices, and confidence interval estimation.

License MIT + file LICENSE

BugReports `https://github.com/cwlee-quantpsych/flex/issues`

Depends R (>= 3.5.0)

Encoding UTF-8

RoxygenNote 7.3.2

Imports ggplot2, plotly, rlang, magrittr

NeedsCompilation no

Author Chaewon Lee [aut, cre],
Jin Hee Yoon [ctb]

Maintainer Chaewon Lee <`chaewon.lee@unc.edu`>

Repository CRAN

Date/Publication 2025-09-02 06:30:07 UTC

Contents

`coefficients` *Define generic for coefficients*

Description

A generic function to retrieve coefficients from model objects.

Usage

```
coefficients(object, ...)
```

Arguments

`object`	The model object from which to extract coefficients.
`...`	Additional arguments (ignored).

Value

A data frame of coefficients and related statistics.

Examples

```
# Simulate data and fit a fuzzy linear model
set.seed(123)
X_crisp <- matrix(round(runif(300, 2, 10)), nrow = 100, ncol = 3)
beta <- c(1.5, -0.8, 2.0)
Y_crisp <- round(X_crisp %*% beta + rnorm(100, mean = 0, sd = 1))
X_fuzzy <- fuzzify_crisp_matrix(X_crisp, spread = 1)
Y_fuzzy <- fuzzify_crisp_vector(Y_crisp, spread = 1)
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 3)

# Extract coefficients
coefficients(object)
```

coefficients.fuzzy_lm *Accessor for Coefficients*

Description

Accessor for Coefficients

Usage

```
## S3 method for class 'fuzzy_lm'
coefficients(object, ...)
```

Arguments

object	An object of class fuzzy_lm.
...	Additional arguments (ignored).

Value

A data frame of coefficients and statistics.

Examples

```
# Simulate data and fit a fuzzy linear model
set.seed(123)
X_crisp <- matrix(round(runif(300, 2, 10)), nrow = 100, ncol = 3)
beta <- c(1.5, -0.8, 2.0)
Y_crisp <- round(X_crisp %*% beta + rnorm(100, mean = 0, sd = 1))
X_fuzzy <- fuzzify_crisp_matrix(X_crisp, spread = 1)
Y_fuzzy <- fuzzify_crisp_vector(Y_crisp, spread = 1)
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 3)

# Extract coefficients
coefficients(object)
```

compute_ci *Compute confidence intervals for regression coefficients*

Description

Compute confidence intervals for regression coefficients

Usage

```
compute_ci(beta_hat, se_beta, df, alpha = 0.05)
```

Arguments

`beta_hat`	Numeric vector. Estimated regression coefficients.
`se_beta`	Numeric vector. Standard errors of coefficients.
`df`	Integer. Degrees of freedom.
`alpha`	Numeric. Significance level (default: 0.05).

Value

A list containing lower and upper bounds of confidence intervals.

Examples

```
beta_hat <- c(0.5, 1.2)  # Example regression coefficients
se_beta <- c(0.1, 0.2)  # Example standard errors
df <- 30  # Example degrees of freedom
ci <- compute_ci(beta_hat, se_beta, df)
print(ci)
```

`compute_pred` *Compute Predictions from Fuzzy Linear Model*

Description

Compute Predictions from Fuzzy Linear Model

Usage

```
compute_pred(object, X_fuzzy)
```

Arguments

`object`	List. Result of fuzzy least squares regression containing `beta_hat`.
`X_fuzzy`	List. Fuzzified predictor variables.

Value

A list of fuzzy predictions.

Examples

```
# Example setup
X_fuzzy <- list(
  list(list(l = 1, x = 2, r = 3), list(l = 4, x = 5, r = 6)),
  list(list(l = 2, x = 3, r = 4), list(l = 5, x = 6, r = 7))
)
beta_hat <- c(0.5, 1.2, -0.8)  # Example regression coefficients
object <- list(beta_hat = beta_hat)
```

```
# Compute predictions
predictions <- compute_pred(object, X_fuzzy)
print(head(predictions, 6))
```

compute_p_values *Compute p-values for regression coefficients*

Description

Compute p-values for regression coefficients

Usage

```
compute_p_values(t_values, df)
```

Arguments

t_values	Numeric vector. T-values for regression coefficients.
df	Integer. Degrees of freedom.

Value

Numeric vector of p-values for each coefficient.

Examples

```
t_values <- c(2.5, 3.0)  # Example t-values
df <- 30  # Example degrees of freedom
p_values <- compute_p_values(t_values, df)
print(p_values)
```

compute_res *Compute Residuals for Fuzzy Linear Model*

Description

Compute Residuals for Fuzzy Linear Model

Usage

```
compute_res(Y_fuzzy, Y_pred)
```

Arguments

Y_fuzzy	List. Fuzzified observed response variables.
Y_pred	List. Fuzzified predicted response variables.

Value

A list of fuzzy residuals.

Examples

```
# Example setup
Y_fuzzy <- list(
  list(l = 2, x = 3, r = 4),
  list(l = 5, x = 6, r = 7)
)
Y_pred <- list(
  list(l = 1.5, x = 2.5, r = 3.5),
  list(l = 4.5, x = 5.5, r = 6.5)
)

# Compute residuals
residuals <- compute_res(Y_fuzzy, Y_pred)
print(head(residuals, 6))
```

`compute_t_values` *Compute t-values for regression coefficients*

Description

Compute t-values for regression coefficients

Usage

```
compute_t_values(beta_hat, Y_fuzzy, Y_pred, XtX_inv)
```

Arguments

`beta_hat`	Numeric vector. Estimated regression coefficients.
`Y_fuzzy`	List. Observed fuzzy responses.
`Y_pred`	List. Predicted fuzzy responses.
`XtX_inv`	Matrix. Inverse of the XtX matrix.

Value

Numeric vector of t-values for the regression coefficients.

Examples

```
# Example setup
beta_hat <- c(0.5, 1.2)  # Example regression coefficients
Y_fuzzy <- list(
  list(l = 2.1, x = 2.3, r = 2.5),
  list(l = 3.1, x = 3.3, r = 3.5),
  list(l = 4.1, x = 4.3, r = 4.5)
)  # Example fuzzy response
Y_pred <- list(
  list(l = 2.0, x = 2.2, r = 2.4),
  list(l = 3.0, x = 3.2, r = 3.4),
  list(l = 4.0, x = 4.2, r = 4.4)
)  # Example predicted values
XtX_inv <- matrix(c(0.1, 0.2, 0.2, 0.4), ncol = 2)  # Example XtX_inv matrix
t_values <- compute_t_values(beta_hat, Y_fuzzy, Y_pred, XtX_inv)
print(t_values)
```

fuzzify_crisp_matrix *Fuzzify a matrix of crisp values*

Description

Converts a numeric matrix into a list of triangular fuzzy numbers.

Usage

```
fuzzify_crisp_matrix(crisp_matrix, spread = 1)
```

Arguments

crisp_matrix	Numeric matrix to be fuzzified.
spread	Numeric. The spread for fuzzification (default is 1).

Value

A list of lists representing rows of triangular fuzzy numbers.

Examples

```
set.seed(123)
matrix <- matrix(runif(9, 5, 15), nrow = 3, ncol = 3)
fuzzify_crisp_matrix(matrix, spread = 1.5)
```

`fuzzify_crisp_value` *Fuzzify a single crisp value*

Description

Converts a crisp value into a triangular fuzzy number with a specified spread.

Usage

```
fuzzify_crisp_value(crisp_value, spread = 1)
```

Arguments

`crisp_value`	Numeric. The crisp value to be fuzzified.
`spread`	Numeric. The spread for fuzzification (default is 1).

Value

A list representing the triangular fuzzy number with components `l`, `x`, and `r`.

Examples

```
fuzzify_crisp_value(10, spread = 2)
```

`fuzzify_crisp_vector` *Fuzzify a vector of crisp values*

Description

Converts a numeric vector into a list of fuzzified values using a triangular fuzzy membership function.

Usage

```
fuzzify_crisp_vector(crisp_vector, spread = 1, var_name = "Outcome")
```

Arguments

`crisp_vector`	A numeric vector to be fuzzified.
`spread`	A non-negative numeric value specifying the spread for the fuzzy membership function.
`var_name`	Optional. A character string specifying a common name for all fuzzified values. Default is `NULL`.

Value

A list of fuzzified values, where each value is represented as a list with components `l`, `x`, and `r`.

Examples

```
crisp_vector <- c(10, 20, 30)
fuzzify_crisp_vector(crisp_vector, spread = 1, var_name = "Variable")
```

`fuzzy_add` *Add two triangular fuzzy numbers*

Description

Performs the addition of two triangular fuzzy numbers.

Usage

```
fuzzy_add(X, Y)
```

Arguments

`X` List. First triangular fuzzy number with components `l`, `x`, and `r`.

`Y` List. Second triangular fuzzy number with components `l`, `x`, and `r`.

Value

A list representing the sum of the two fuzzy numbers.

Examples

```
X <- list(l = 1, x = 2, r = 3)
Y <- list(l = 2, x = 3, r = 4)
fuzzy_add(X, Y)
```

`fuzzy_crisp_mult` *Multiply a crisp scalar by a triangular fuzzy number*

Description

Scales a triangular fuzzy number by a crisp scalar.

Usage

```
fuzzy_crisp_mult(scalar, fuzzy_num)
```

Arguments

scalar	Numeric. The scalar to multiply with the fuzzy number.
fuzzy_num	List. A triangular fuzzy number with components l, x, and r.

Value

A list representing the scaled fuzzy number.

Examples

```
scalar <- 3
fuzzy_num <- list(l = 1, x = 2, r = 3)
fuzzy_crisp_mult(scalar, fuzzy_num)
```

fuzzy_d_squared *Compute the squared distance between two fuzzy numbers*

Description

Calculates the squared distance between two triangular fuzzy numbers using Diamond's metric.

Usage

```
fuzzy_d_squared(X, Y)
```

Arguments

X	List. First triangular fuzzy number.
Y	List. Second triangular fuzzy number.

Value

Numeric. The squared distance between X and Y.

Examples

```
X <- list(l = 1, x = 2, r = 3)
Y <- list(l = 2, x = 3, r = 4)
fuzzy_d_squared(X, Y)
```

`fuzzy_lm` *Fuzzy Linear Regression*

Description

Fits a fuzzy linear regression model given fuzzified predictors and response variables.

Usage

```
fuzzy_lm(X_fuzzy, Y_fuzzy, p, X_crisp = NULL)
```

Arguments

`X_fuzzy` A list of fuzzified predictor values.

`Y_fuzzy` A list of fuzzified response values.

`p` An integer specifying the number of predictors.

`X_crisp` Optional. The original crisp predictor matrix or data frame. Used to retrieve variable names. Default is `NULL`.

Value

A list object of class `fuzzy_lm` containing:

`Coefficients` A data frame with estimated coefficients, standard errors, t-values, p-values, and significance stars.

`Residuals` The residuals from the fitted model.

`Predictions` The predicted fuzzified response values.

`RSS` The residual sum of squares.

`R_squared` The coefficient of determination (R-squared).

`Sigma_squared` The estimated variance of the residuals.

`Degrees_of_Freedom` The degrees of freedom for the model.

Examples

```
# Simulate complex data for fuzzy linear regression
set.seed(123)

# Generate a dataset with 100 observations and 4 predictors
n <- 100
X_crisp <- data.frame(
  Age = round(runif(n, 20, 70)),            # Random ages between 20 and 70
  Income = round(runif(n, 20000, 120000)),  # Random incomes between 20k and 120k
  Education = round(runif(n, 10, 20)),      # Random years of education between 10 and 20
  Experience = round(runif(n, 1, 40))     # Random years of work experience between 1 and 40
)
```

```
# Define true coefficients
beta <- c(5.0, 1.2, -0.5, 0.8, 0.05)  # Intercept and coefficients for the predictors

# Generate the crisp response variable with noise
Y_crisp <- round(beta[1] + as.matrix(X_crisp) %*% beta[-1] + rnorm(n, mean = 0, sd = 50))

# Fuzzify the predictor and response variables
X_fuzzy <- fuzzify_crisp_matrix(as.matrix(X_crisp), spread = 10)  # Larger spread for predictors
Y_fuzzy <- fuzzify_crisp_vector(Y_crisp, spread = 20)         # Larger spread for responses

# Fit the fuzzy linear model
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 4, X_crisp = X_crisp)

# Print the coefficients
print("Fuzzy Linear Model Coefficients:")
print(object$Coefficients)

# Example residuals and predictions
print("Example Residuals:")
print(head(object$Residuals, 6))

print("Example Predictions:")
print(head(object$Predictions, 6))
```

fuzzy_mults *Multiply two triangular fuzzy numbers*

Description

Computes the scalar product of two triangular fuzzy numbers.

Usage

```
fuzzy_mults(X, Y)
```

Arguments

X	List. First triangular fuzzy number with components l, x, and r.
Y	List. Second triangular fuzzy number with components l, x, and r.

Value

A scalar representing the sum of the product of the corresponding components.

Examples

```
X <- list(l = 1, x = 2, r = 3)
Y <- list(l = 2, x = 3, r = 4)
fuzzy_mults(X, Y)
```

plot *Generic Plot Function*

Description

This is a generic plot function that dispatches to specific plot methods based on the class of the object provided. It is used to create plots for objects such as fuzzy_lm.

Usage

```
plot(object, ...)
```

Arguments

object	The object to be plotted.
...	Additional arguments passed to specific plot methods.

Value

Depends on the class of object. Typically, a plot or visualization is returned.

Examples

```
# Example with fuzzy_lm:
set.seed(123)
x_crisp <- seq(4, 12, length.out = 20)
beta <- 1.5
intercept <- 2
y_crisp <- intercept + beta * x_crisp + rnorm(length(x_crisp), mean = 0, sd = 0.5)

# Fuzzify data
spread_x <- 0.5
spread_y <- 1.0
X_fuzzy <- fuzzify_crisp_matrix(matrix(x_crisp, ncol = 1), spread = spread_x)
Y_fuzzy <- fuzzify_crisp_vector(y_crisp, spread = spread_y)

# Fit fuzzy regression model
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 1)

# Plot

plot(object, X_fuzzy = X_fuzzy, Y_fuzzy = Y_fuzzy)
```

plot.fuzzy_lm *Plot Fuzzy Regression Results*

Description

Visualizes the results of a fuzzy regression model. For simple regression (1 predictor), it generates a 2D plot with fuzzy intervals and regression lines. For multiple regression (2 predictors), it generates a 3D plot with cubes representing fuzzy intervals and a regression plane.

Usage

```
## S3 method for class 'fuzzy_lm'
plot(object, ...)
```

Arguments

object	An object of class fuzzy_lm.
...	Additional arguments passed to the method, including: • X_fuzzy: A list of fuzzified predictor variables. • Y_fuzzy: A list of fuzzified outcome variables.

Value

A ggplot2 object for simple regression or a plotly object for multiple regression.

Examples

```
# Example 1: Simple Regression
# See above for setup example

# Example 2: Multiple Regression
set.seed(123)
n <- 100
x1_crisp <- runif(n, 5, 15)
x2_crisp <- runif(n, 10, 20)
beta <- c(3, 1.5, -0.8)
y_crisp <- beta[1] + beta[2] * x1_crisp + beta[3] * x2_crisp + rnorm(n, mean = 0, sd = 2)

X_fuzzy <- fuzzify_crisp_matrix(cbind(x1_crisp, x2_crisp), spread = 0.5)
Y_fuzzy <- fuzzify_crisp_vector(y_crisp, spread = 1.0)
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 2)

plot(object, X_fuzzy = X_fuzzy, Y_fuzzy = Y_fuzzy)
```

`predictions` *Define generic for predictions*

Description

Define generic for predictions

Usage

```
predictions(object, ...)
```

Arguments

`object`	An object of class `fuzzy_lm`. The model object.
`...`	Additional arguments (currently ignored).

Value

A list of fuzzy predictions.

Examples

```
# Simulate data and fit a fuzzy linear model
set.seed(123)
X_crisp <- matrix(round(runif(300, 2, 10)), nrow = 100, ncol = 3)
beta <- c(1.5, -0.8, 2.0)
Y_crisp <- round(X_crisp %*% beta + rnorm(100, mean = 0, sd = 1))
X_fuzzy <- fuzzify_crisp_matrix(X_crisp, spread = 1)
Y_fuzzy <- fuzzify_crisp_vector(Y_crisp, spread = 1)
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 3)

# Extract predictions
head(predictions(object))
```

`predictions.fuzzy_lm` *Accessor for Predictions*

Description

Accessor for Predictions

Usage

```
## S3 method for class 'fuzzy_lm'
predictions(object, ...)
```

Arguments

`object`	An object of class `fuzzy_lm`. The model object.
`...`	Additional arguments (currently ignored).

Value

A list of fuzzy predictions.

Examples

```
# Simulate data and fit a fuzzy linear model
set.seed(123)
X_crisp <- matrix(round(runif(300, 2, 10)), nrow = 100, ncol = 3)
beta <- c(1.5, -0.8, 2.0)
Y_crisp <- round(X_crisp %*% beta + rnorm(100, mean = 0, sd = 1))
X_fuzzy <- fuzzify_crisp_matrix(X_crisp, spread = 1)
Y_fuzzy <- fuzzify_crisp_vector(Y_crisp, spread = 1)
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 3)

# Extract predictions
head(predictions(object))
```

`residuals` *Define generic for residuals*

Description

A generic function to retrieve residuals from model objects.

Usage

```
residuals(object, ...)
```

Arguments

`object`	The model object from which to extract residuals.
`...`	Additional arguments (ignored).

Value

A list of fuzzy residuals.

Examples

```
# Simulate data and fit a fuzzy linear model
set.seed(123)
X_crisp <- matrix(round(runif(300, 2, 10)), nrow = 100, ncol = 3)
beta <- c(1.5, -0.8, 2.0)
Y_crisp <- round(X_crisp %*% beta + rnorm(100, mean = 0, sd = 1))
X_fuzzy <- fuzzify_crisp_matrix(X_crisp, spread = 1)
Y_fuzzy <- fuzzify_crisp_vector(Y_crisp, spread = 1)
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 3)

# Extract residuals
head(residuals(object))
```

---

residuals.fuzzy_lm *Accessor for Residuals*

Description

Accessor for Residuals

Usage

```
## S3 method for class 'fuzzy_lm'
residuals(object, ...)
```

Arguments

`object`	An object of class `fuzzy_lm`. The model object.
`...`	Additional arguments (currently ignored).

Value

A list of fuzzy residuals.

Examples

```
# Simulate data and fit a fuzzy linear model
set.seed(123)
X_crisp <- matrix(round(runif(300, 2, 10)), nrow = 100, ncol = 3)
beta <- c(1.5, -0.8, 2.0)
Y_crisp <- round(X_crisp %*% beta + rnorm(100, mean = 0, sd = 1))
X_fuzzy <- fuzzify_crisp_matrix(X_crisp, spread = 1)
Y_fuzzy <- fuzzify_crisp_vector(Y_crisp, spread = 1)
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 3)

# Extract residuals
head(residuals(object))
```

summary.fuzzy_lm *Summary for Fuzzy Linear Regression*

Description

Summary for Fuzzy Linear Regression

Usage

```
## S3 method for class 'fuzzy_lm'
summary(object, ...)
```

Arguments

object	An object of class fuzzy_lm. The model object.
...	Additional arguments (currently ignored).

Value

Prints a summary of the fuzzy linear regression results.

Examples

```
# Simulate data and fit a fuzzy linear model
set.seed(123)
X_crisp <- matrix(round(runif(300, 2, 10)), nrow = 100, ncol = 3)
beta <- c(1.5, -0.8, 2.0)
Y_crisp <- round(X_crisp %*% beta + rnorm(100, mean = 0, sd = 1))
X_fuzzy <- fuzzify_crisp_matrix(X_crisp, spread = 1)
Y_fuzzy <- fuzzify_crisp_vector(Y_crisp, spread = 1)
object <- fuzzy_lm(X_fuzzy, Y_fuzzy, p = 3)

# Summarize the model
summary(object)
```

Index