Package 'QuantBondCurves'

May 7, 2026

Type Package

Title Calculates Bond Values and Interest Rate Curves for Finance

Version 0.3.2

Maintainer Camilo Díaz <kamodiaz@gmail.com>

Description Values different types of assets and calibrates discount curves
for quantitative financial analysis. It covers fixed coupon assets,
floating note assets, interest and cross currency swaps with different
payment frequencies. Enables the calibration of spot, instantaneous forward
and basis curves, making it a powerful tool for accurate and flexible bond
valuation and curve generation. The valuation and calibration techniques
presented here are consistent with industry standards and incorporates
author's own calculations. Tuckman, B., Serrat, A. (2022, ISBN: 978-1-119-83555-4).

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.1

Imports lubridate, quantdates, Rsolnp

Depends R (>= 3.5.0)

Suggests testthat (>= 3.0.0), knitr, rmarkdown, ggplot2

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Camilo Díaz [aut, cre, com],
Andrés Galeano [aut],
Julián Rojas [aut],
Quantil S.A.S [aut, cph]

Repository CRAN

Date/Publication 2025-07-06 07:50:06 UTC

Contents

`accrued.interests` *Accrued interest*

Description

Calculates the accumulated coupon or accrued interests of the asset, from its last coupon or cash flow payment.

Usage

```
accrued.interests(
  maturity,
  analysis.date = Sys.Date(),
  coupon.rate,
  principal = 1,
  asset.type = "TES",
  freq = NULL,
  daycount = "ACT/360"
)
```

Arguments

`maturity`	Last day of the contract: YYYY-MM-DD. Alternatively, it can be a numeric value that represents the duration of the contract in years.
`analysis.date`	Date in which the asset is valued. By default, the current date.
`coupon.rate`	Coupon rate of the asset. Can be an unique numeric value or a vector corresponding to each coupon payment date.

principal	Notional amount for the asset.
asset.type	String that determines the asset type to value. See also 'Details'.
freq	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
daycount	Day count convention. See also 'Details'.

Details

asset.type makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.
- "IBR" for bonds and assets indexed to 3M IBR rate.
- "IBRSwaps" for swaps indexed to IBR rate.
- "LIBOR" for bonds and assets indexed to 3M LIBOR.
- "UVRSwaps" for cross-currency swaps indexed to UVR-IBR rate.
- "LIBORSwaps" for Interest Rate Swaps (IRS) indexed to 3M LIBOR.

daycount convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.
- ACT/ACT-ISDA
- ACT/ACT-AFB

Value

Accrued interest of the bond from the last coupon payment to the liquidation (valuation date).

Examples

```
accrued.interests(coupon.rate = 0.04, maturity = '2029-08-10',
                  asset.type = 'LIBOR', daycount = "30/360")
accrued.interests(coupon.rate = 0.04, maturity = '2029-08-10',
                  daycount = "NL/365")
accrued.interests(coupon.rate = 0.04, maturity = '2029-08-10',
                  asset.type= 'IBR', daycount = "ACT/360")
accrued.interests(coupon.rate = 0.04, maturity = '2029-08-10', freq= 2,
                  asset.type= 'FixedIncome', daycount = "ACT/365")
```

`average.life` *Weighted Average Life*

Description

Calculates the weighted average life of a given bond by dividing the weighted total payments by the total payments.

Usage

```
average.life(
  input,
  price,
  maturity,
  analysis.date = Sys.Date(),
  coupon.rate,
  principal = 1,
  asset.type = "TES",
  freq = 1,
  rate.type = 1,
  spread = 0,
  daycount = "ACT/365",
  dirty = 1,
  convention = "F",
  trade.date = NULL,
  coupon.schedule = "SF"
)
```

Arguments

`input`	String that establishes if the price input corresponds to the Internal Rate of Return (IRR) of the bond or the market price. Set `"rate"` for the IRR. Otherwise, `"price"`.
`price`	Numeric value of either market price or Internal Rate of Return (IRR) of a given bond. Instead of IRR, can also be a rates vector that corresponds to coupon dates.
`maturity`	Last day of the contract: YYYY-MM-DD. Alternatively, it can be a numeric value that represents the duration of the contract in years.
`analysis.date`	Date in which the asset is valued. By default, the current date.
`coupon.rate`	Coupon rate of the asset. Can be an unique numeric value or a vector corresponding to each coupon payment date.
`principal`	Notional amount for the asset.
`asset.type`	String that determines the asset type to value. See also 'Details'.
`freq`	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).

rate.type	(1) for discrete compounded discount rates and (0) for continuosly compounded discount rates. By default rates are assumed to be discrete.
spread	Decimal value of spread added to coupon payment rate. By default, 0.
daycount	Day count convention. See also 'Details'.
dirty	Numeric value to determine if the calculated price is dirty or clean. To calculate dirty price, set dirty = 1. Otherwise, dirty = 0.
convention	String that establishes if the effective dates are calculated using Following, Modified Following, Backward or Backward Following. See also 'Details'.
trade.date	The date on which the transaction occurs. It is used to calculate maturity as a date, when given in years. Also required for non-trivial cases such as bonds with long first coupon.
coupon.schedule	String that establishes if a bond first coupon period is a long first coupon or a short first coupon. On the contrary, establishes if last coupon period is long last coupon or a short last coupon. See also 'Details'.

Details

asset.type makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.
- "IBR" for bonds and assets indexed to 3M IBR rate.
- "LIBOR" for bonds and assets indexed to 3M LIBOR.

daycount convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.
- ACT/ACT-ISDA
- ACT/ACT-AFB

convention makes reference to the following type of business day conventions:

- "F" for Following business day convention.
- "MF" for Modified Following business day convention.
- "B" for Backward business day convention.
- "MB" for Modified Backward business day convention.

coupon.schedule makes reference to the following type of coupon payment schedule of a bond:

- "LF" for Long First coupon payment.
- "LL" for Long Last coupon payment.
- "SF" for Short First coupon payment.
- "SL" for Short Last coupon payment.

Value

Weighted average life of given bond

Examples

```
average.life(input = c("rate"), price = 0.08, maturity = "2026-06-01",
             analysis.date = "2025-06-01", coupon.rate = 0.06, principal = 1000,
             asset.type = "IBR", freq = 4)
average.life(input = c("rate"), price = c(0.043,0.05), maturity = "2023-01-03",
             analysis.date = "2021-01-03", coupon.rate = 0.04, principal = 1,
             asset.type = "FixedIncome", freq = 1, rate.type = 0)
```

---

basis.curve *Basis Curve*

Description

Function that calibrates a "discount basis rate" curve according to data of cross currency swaps. Available methods are bootstrapping or residual sum of squares (RSS) between the value of a given or inferred fixed foreign leg and the value of the local leg.

Usage

```
basis.curve(
  swaps,
  ex.rate = NULL,
  analysis.date = Sys.Date(),
  rates,
  rates2,
  freq = 1,
  rate.type = 1,
  daycount = "ACT/365",
  npieces = NULL,
  obj = "Price",
  Weights = NULL,
  nsimul = 1,
  piece.term = NULL,
  nodes = seq(0, 15, 0.001),
  approximation = "constant"
)
```

Arguments

swaps	Matrix containing relevant information of cross currency swaps where each row represents a swap and each column represents the next attributes: maturity, legs, coupon rate of local leg, coupon rate of foreign leg, spread of local leg, spread of

	variable leg, principal of local and principal of variable leg. `colnames(swaps)` must be defined as the following: `c("Mat", "Legs", "C1", "C2", "spread1", "spread2", "prin1", "prin2")`.
`ex.rate`	Exchange rate on analysis date. Format has to be local currency divided by foreign currency.
`analysis.date`	Date in which the asset is valued. By default, the current date.
`rates`	Discount rates given by the local zero coupon rate curve. The curve has to have nodes with at least, with 3 decimals.
`rates2`	Discount rates given by the foreign zero coupon rate curve. The curve has to have nodes with at least, with 3 decimals.
`freq`	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
`rate.type`	(1) for discrete compounded discount rates and (0) for continuosly compounded discount rates. By default rates are assumed to be discrete.
`daycount`	Day count convention. See also 'Details'.
`npieces`	Number of constant or linear segments for the curve to have. By default `NULL`, and bootstrapping method is used, otherwise, minimization of RSS is used.
`obj`	String related to the definition of the error in the RSS methodology. Set `"Price"` for minimization of error by price or `"Rate"` for minimization of error by rate.
`Weights`	Vector of weights used to dot product with residual squares in order to calculate residual sum of squares. By default, each residual is assigned the same weight.
`nsimul`	Number of simulations for the terms of the pieces. The more simulations, the more likely to find a better local solution. By default `1`, and terms are defined in such way each piece occupies the same length in the abscissa axis.
`piece.term`	Vector that establishes a unique term structure for optimization to take place. Each piece or segment must have a unique maturity, as numeric value in years, that signifies the end of the segment. Last segment maturity must not be introduced, it is assumed to be equivalent to the last term introduced on analysis date. Therefore, the `piece.term` vector must always have a length equal to `npieces - 1`.
`nodes`	Desired output nodes of the curve.
`approximation`	String that establish the approximation. Set `'linear'` for a linear approximation, or `'constant'` for a piecewise constant function.

Details

`daycount` convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.

- ACT/ACT-ISDA
- ACT/ACT-AFB

`swaps["Legs"]` makes reference to the following types of legs composition of the cross currency swaps.

- "FF" for fixed leg in local currency and fixed leg in foreign currency.
- "FV" for fixed leg in local currency and variable leg in foreign currency.
- "VF" for variable leg in local currency and fixed leg in foreign currency.
- "VV" for variable leg in local currency and variable leg in foreign currency.

Value

Constant or Linear piecewise basis curve.

Author(s)

Camilo Díaz

Examples

```
# Inputs for calibration of spot curve
yield.curve <- c(0.015,0.0175, 0.0225, 0.0275, 0.0325, 0.0375,0.04,0.0425,0.045,0.0475,0.05)
names(yield.curve) <- c(0.5,1,2,3,4,5,6,7,8,9,10)
nodes <- seq(0,10,0.001)
# Calibration of local spot curve
rates <- curve.calibration (yield.curve = yield.curve, market.assets = NULL,
                            analysis.date = "2019-01-03" , asset.type = "IBRSwaps",
                            freq = 4, rate.type = 0, fwd = 0, npieces = NULL,
                            obj = "Price", nodes = nodes, approximation = "linear")
# Input for Basis Curve
ex.rate <- 4814
swaps <- rbind(c("2024-03-01", "FF", 0.07 , 0.0325, NA   , NA    , 2000 * ex.rate, 2000),
               c("2025-03-01", "VV", NA   , NA    , 0.015, 0.0175, 2000 * ex.rate, 2000),
            c("2026-03-01", "FF", 0.075, 0.03  , NA   ,  NA   , 5000000, 5000000 / ex.rate),
            c("2027-03-01", "VV", NA   , NA    , 0.01 , 0.015 , 5000000, 5000000 / ex.rate),
            c("2028-03-01", "FF", 0.08 ,0.035  , NA   , NA    , 3000000, 3000000 / ex.rate),
            c("2029-03-01", "VV", NA   , NA    , 0.01 , 0.0125, 3000000, 3000000 / ex.rate))
colnames(swaps) <- c("Mat"  ,"Legs", "C1" , "C2", "spread1", "spread2", "prin1", "prin2")
# Function
basis.curve(swaps = swaps, ex.rate = 4814, analysis.date = "2023-03-01",
            rates = rates, rates2 = rates / 4, freq = c(2,2,2,2,1,1),
            rate.type = 1, npieces = 4, obj = "Price", Weights = NULL,
            nsimul = 1, nodes = nodes, approximation = "linear")
```

`bond.price2rate` *From a price to a rate*

Description

Calculates the Internal Rate of Return (IRR) of a given asset taking into account the market price, maturity, face value, and analysis date.

Usage

```
bond.price2rate(
  maturity,
  analysis.date = Sys.Date(),
  price,
  coupon.rate,
  principal = 1,
  asset.type = "TES",
  freq = NULL,
  rate.type = 1,
  spread = 0,
  dirty = 1,
  daycount = "NL/365",
  convention = "F",
  trade.date = NULL,
  coupon.schedule = "SF"
)
```

Arguments

`maturity`	Last day of the contract: YYYY-MM-DD. Alternatively, it can be a numeric value that represents the duration of the contract in years.
`analysis.date`	Date in which the asset is valued. By default, the current date.
`price`	Numeric value. Price of the bond to convert.
`coupon.rate`	Coupon rate of the asset. Can be an unique numeric value or a vector corresponding to each coupon payment date.
`principal`	Notional amount for the asset.
`asset.type`	String that determines the asset type to value. See also 'Details'.
`freq`	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
`rate.type`	(1) for discrete compounded discount rates and (0) for continuosly compounded discount rates. By default rates are assumed to be discrete.
`spread`	Decimal value of spread added to coupon payment rate. By default, `0`.
`dirty`	Numeric value to determine if the calculated price is dirty or clean. To calculate dirty price, set `dirty = 1`. Otherwise, `dirty = 0`.

daycount	Day count convention. See also 'Details'.
convention	String that establishes if the effective dates are calculated using Following, Modified Following, Backward or Backward Following. See also 'Details'.
trade.date	The date on which the transaction occurs. It is used to calculate maturity as a date, when given in years. Also required for non-trivial cases such as bonds with long first coupon.
coupon.schedule	String that establishes if a bond first coupon period is a long first coupon or a short first coupon. On the contrary, establishes if last coupon period is long last coupon or a short last coupon. See also 'Details'.

Details

`asset.type` makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.
- "IBR" for bonds and assets indexed to 3M IBR rate.
- "LIBOR" for bonds and assets indexed to 3M LIBOR.

`daycount` convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.
- ACT/ACT-ISDA
- ACT/ACT-AFB

`convention` makes reference to the following type of business day conventions:

- "F" for Following business day convention.
- "MF" for Modified Following business day convention.
- "B" for Backward business day convention.
- "MB" for Modified Backward business day convention.

`coupon.schedule` makes reference to the following type of coupon payment schedule of a bond:

- "LF" for Long First coupon payment.
- "LL" for Long Last coupon payment.
- "SF" for Short First coupon payment.
- "SL" for Short Last coupon payment.

Value

The Yield to Maturity or Internal Rate of Return of a given asset.

Examples

```
bond.price2rate(maturity = "2023-01-03", analysis.date = "2021-01-03",
                price = 1, coupon.rate = 0.04, principal = 1,
                asset.type = "TES", freq = 1)
```

---

coupon.dates *Coupon date payments*

Description

Function to calculate the upcoming coupon payment dates of a given asset, based on its payment frequency. The list of payment dates encompass the time period between the analysis date and the maturity of the asset.

Usage

```
coupon.dates(
  maturity,
  analysis.date = Sys.Date(),
  asset.type = "TES",
  freq = NULL,
  convention = "F",
  loc = "BOG",
  trade.date = NULL,
  coupon.schedule = "SF"
)
```

Arguments

maturity	Last day of the contract: YYYY-MM-DD. Alternatively, it can be a numeric value that represents the duration of the contract in years.
analysis.date	Date in which the asset is valued. By default, the current date.
asset.type	String that determines the asset type to value. See also 'Details'.
freq	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
convention	String that establishes if the effective dates are calculated using Following, Modified Following, Backward or Backward Following. See also 'Details'.
loc	String related to the location of the asset. It is used to calculate the effective dates, taking into account the business days of the given location. See also 'Details'.

`trade.date` The date on which the transaction occurs. It is used to calculate maturity as a date, when given in years. Also required for non-trivial cases such as bonds with long first coupon.

`coupon.schedule`
String that establishes if a bond first coupon period is a long first coupon or a short first coupon. On the contrary, establishes if last coupon period is long last coupon or a short last coupon. See also 'Details'.

Details

`asset.type` makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.
- "IBR" for bonds and assets indexed to 3M IBR rate.
- "IBRSwaps" for swaps indexed to IBR rate.
- "LIBOR" for bonds and assets indexed to 3M LIBOR.
- "UVRSwaps" for cross-currency swaps indexed to UVR-IBR rate.
- "LIBORSwaps" for Interest Rate Swaps (IRS) indexed to 3M LIBOR.

`loc` makes reference to the following locations:

- "BOG" for colombian issued assets and national business days (default).
- "LDN" for business days of London.
- "NY" for business days of New York
- "NYLDN" for the intersection of business days in New York and London.
- "BOGNY" for the intersection of business days in Bogota and New York.

`convention` makes reference to the following type of business day conventions:

- "F" for Following business day convention.
- "MF" for Modified Following business day convention.
- "B" for Backward business day convention.
- "MB" for Modified Backward business day convention.

`coupon.schedule` makes reference to the following type of coupon payment schedule of a bond:

- "LF" for Long First coupon payment.
- "LL" for Long Last coupon payment.
- "SF" for Short First coupon payment.
- "SL" for Short Last coupon payment.

Value

Upcoming coupon dates and dates of payment according to business day conventions.

Note

If only maturity is given, function assumes that the coupon payments have already started. If `maturity` and `trade.date` are included, coupon dates are calculated from `trade.date` to `maturity`. If by doing so, `trade.date` doesn't converge to `maturity`, month remainder is adjusted according to `coupon.schedule`. For LIBOR assets, function adds 2 business days to `trade.date`.

Examples

```
coupon.dates("2028-04-03")
coupon.dates(maturity = 2, analysis.date = "2021-10-01")
coupon.dates(maturity = "2029-10-01", asset.type = "FixedIncome", freq = 2, convention = "MB")
coupon.dates(maturity = "2028-02-29", analysis.date = "2022-07-29", trade.date = "2022-07-29",
             asset.type = "TES", coupon.schedule = "SF")
coupon.dates(maturity = "2025-11-30", analysis.date = "2022-03-01", trade.date = "2021-05-31",
             asset.type = "IBR", loc = "NY", convention = "F")
```

---

coupons *Coupon payment calculation*

Description

Function that returns coupon values according to specified payment dates and a day count convention. Yields the values of cash flows for the remaining duration of assets, following a date payment structure, face value -or principal- and a specified coupon rate.

Usage

```
coupons(
  dates = NULL,
  coupon.rate,
  principal = 1,
  asset.type = "TES",
  freq = NULL,
  daycount = "ACT/360",
  loc = "BOG",
  maturity = NULL,
  analysis.date = Sys.Date(),
  trade.date = NULL,
  coupon.schedule = "SF"
)
```

Arguments

`dates`	Coupon payment dates.
`coupon.rate`	Coupon rate of the asset. Can be an unique numeric value or a vector corresponding to each coupon payment date.
`principal`	Notional amount for the asset.

asset.type	String that determines the asset type to value. See also 'Details'.
freq	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
daycount	Day count convention. See also 'Details'.
loc	String related to the location of the asset. It is used to calculate the effective dates, taking into account the business days of the given location. See also 'Details'.
maturity	Only necessary in cases where coupon payment dates are not provided in the dates parameter. Last day of the contract. Can be a numeric value that represents the duration of the contract in years.
analysis.date	Date in which the asset is valued. By default, the current date.
trade.date	The date on which the transaction occurs. It is used to calculate maturity as a date, when given in years. Also required for non-trivial cases such as bonds with long first coupon.
coupon.schedule	String that establishes if a bond first coupon period is a long first coupon or a short first coupon. On the contrary, establishes if last coupon period is long last coupon or a short last coupon. See also 'Details'.

Details

asset.type makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.
- "IBR" for bonds and assets indexed to 3M IBR rate.
- "IBRSwaps" for swaps indexed to IBR rate.
- "LIBOR" for bonds and assets indexed to 3M LIBOR.
- "UVRSwaps" for cross-currency swaps indexed to UVR-IBR rate.
- "LIBORSwaps" for Interest Rate Swaps (IRS) indexed to 3M LIBOR.

daycount convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.
- ACT/ACT-ISDA
- ACT/ACT-AFB

coupon.schedule makes reference to the following type of coupon payment schedule of a bond:

- "LF" for Long First coupon payment.
- "LL" for Long Last coupon payment.
- "SF" for Short First coupon payment.
- "SL" for Short Last coupon payment.

Value

$Coupons or $Cash flows of the asset in analysis.

Examples

```
coupons(dates = c("2020-09-10", "2020-12-10", "2021-03-10"),
        coupon.rate = 0.06)
coupons(dates = c("2020-09-10", "2020-12-10", "2021-03-10"),
        coupon.rate = 0.08, principal = 1000,
        asset.type = "LIBOR", daycount = "ACT/365")
coupons(dates = c("2020-09-10","2020-12-10", "2021-03-10"),
        coupon.rate = 0.07, asset.type = "FixedIncome",
        freq = 4, daycount = "NL/365")
coupons(coupon.rate = c(0.04,0.04,0.42,0.045,0.05),
        maturity = "2024-01-05", analysis.date = "2023-01-03",
        asset.type = "IBR")
coupons(coupon.rate = 0.03, maturity = 1.08,
        analysis.date = "2020-02-29", trade.date = "2020-02-29",
        asset.type = "IBR", coupon.schedule = "LF")
```

---

curve.calculation *Curve calculation*

Description

Function that calculates zero coupon or instantaneous forward curves for multiple analysis dates, according to historical data of internal rates of return (IRR) and coupon rates of assets. Extends previous market rate curves by minimizing Mean Absolute Errors (MAE) following a bootstrapping recursive method. Alternatively, methodology of residual sum of squares (RSS) can be employed.

Usage

```
curve.calculation(
  serie,
  market.assets = NULL,
  noSpots = 1,
  previous.curve = NULL,
  asset.type = "TES",
  freq = 1,
  rate.type = 1,
  daycount = NULL,
```

```
  fwd = 0,
  npieces = NULL,
  obj = "Price",
  Weights = NULL,
  nsimul = 1,
  piece.term = NULL,
  nodes = seq(0, 10, 0.001),
  approximation = "constant"
)
```

Arguments

serie	A time series matrix that encompasses a sequence of IRR's emitted on distinct dates. The columns correspond to different maturity periods, expressed in years, while the row names indicate the precise dates when the rates were emitted.
market.assets	A matrix containing market assets data, where the first column represents the coupon rates of the assets and the second column represents their corresponding maturities as dates. This input is required only if the IRR's of assets differs from their coupon rates.
noSpots	Number of spot interest rates introduced in the serie input per row. Function assumes spots are the first entries on the serie vector for every row.
previous.curve	A matrix that stores historical curve values up to the earliest calibration date on serie. The row names correspond to the dates on which the curve was calculated, while the columns represent maturities as years.
asset.type	String that determines the asset type to value. See also 'Details'.
freq	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
rate.type	(1) for annual compounded discount rates and (0) for continuosly compounded discount rates. By default, rates are assumed to be the former.
daycount	Day count convention. See also 'Details'.
fwd	Numeric value that determines if the desired output curve is a forward or a spot curve. Set 0 for spot curve (default), 1 otherwise.
npieces	Number of constant or linear segments for the curve to incorporate. By default NULL, and bootstrapping method is used, otherwise, minimization of RSS is used.
obj	String related to the definition of the error in the RSS methodology. Set "Price" for minimization of error by price or "Rate" for minimization of error by rate.
Weights	Vector of weights used to dot product with residual squares in order to calculate residual sum of squares. By default, each residual is assigned the same weight.
nsimul	Number of simulations for the terms of the pieces. The more simulations, the more likely to find a better local solution. By default 1, and terms are defined in such way each piece occupies the same length in the abscissa axis.

piece.term	Vector that establishes a unique term structure for optimization to take place. Each piece or segment must have a unique maturity, as numeric value in years, that signifies the end of the segment. Last segment maturity must not be introduced, it is assumed to be equivalent to the last term introduced on analysis date. Therefore, the piece.term vector must always have a length equal to npieces - 1.
nodes	Desired output nodes of the curve.
approximation	String that establish the approximation. Set 'linear' for a piecewise linear approximation, or 'constant' for a piecewise constant curve.

Details

asset.type makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "IBRSwaps" for swaps indexed to IBR rate.
- "LIBORSwaps" for Interest Rate Swaps (IRS) indexed to 3M LIBOR.
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.

If npieces = NULL uses a recursive iteration process based in bootstrapping where the curve is constructed through a minimization of the MAE between the dirty price of historical assets and an approximation of the theoretical price of assets of same maturity. Uses the "L-BFGS-B" optimization method to minimize the expected MAE. Otherwise, curve is constructed through minimization of RSS where the error can be defined via price or rate.

Value

Zero Coupon curves for the corresponding analysis dates. If fwd = 1, returns forward curves.

Examples

```
# Previous curve input
previous.curve <- matrix(0.04,nrow = 2,ncol = 8)
rownames(previous.curve) <- c("2014-01-01","2015-01-01")
colnames(previous.curve) <- c(0, 0.25, 0.5, 1:5)
# IRR's input
serie <- matrix(NA,nrow = 4,ncol = 6)
rownames(serie) <- c("2014-01-01","2015-01-01","2016-01-01","2017-01-01")
colnames(serie) <- c(0, 0.08333, 0.25, 0.5, 1, 2)
serie[1,1] <- 0.040; serie[1,2] <- 0.050; serie[1,3] <- 0.060; serie[1,4] <- 0.065
serie[1,5] <- 0.070; serie[1,6] <- 0.075
serie[2,1] <- 0.030; serie[2,2] <- 0.040; serie[2,3] <- 0.050; serie[2,4] <- 0.063
serie[2,5] <- 0.074; serie[2,6] <- 0.080
serie[3,1] <- 0.060; serie[3,2] <- 0.065; serie[3,3] <- 0.070; serie[3,4] <- 0.080
serie[3,5] <- 0.084; serie[3,6] <- 0.090
serie[4,1] <- 0.020; serie[4,2] <- 0.030; serie[4,3] <- 0.040; serie[4,4] <- 0.042
serie[4,5] <- 0.045; serie[4,6] <- 0.050
# Market Assets input
market.assets <- matrix(NA,nrow = 10,ncol = 2)
```

```
market.assets[1,1]  <- 0.040 ; market.assets[2,1]  <- 0.05
market.assets[3,1]  <- 0.060 ; market.assets[4,1]  <- 0.07
market.assets[5,1]  <- 0.080 ; market.assets[6,1]  <- 0.09
market.assets[7,1]  <- 0.060 ; market.assets[8,1]  <- 0.07
market.assets[9,1]  <- 0.075 ; market.assets[10,1] <- 0.07
market.assets[1,2]  <- "2016-01-01" ; market.assets[2,2]  <- "2016-02-01"
market.assets[3,2]  <- "2016-04-01" ; market.assets[4,2]  <- "2016-07-01"
market.assets[5,2]  <- "2017-01-01" ; market.assets[6,2]  <- "2017-02-01"
market.assets[7,2]  <- "2017-04-01" ; market.assets[8,2]  <- "2017-07-01"
market.assets[9,2]  <- "2018-01-01" ; market.assets[10,2] <- "2019-01-01"
#Calculation
curve.calculation(serie = serie, market.assets = market.assets,
                  previous.curve = previous.curve, asset.type = "TES",
                  freq = 1, rate.type = 1, fwd = 0,
                  nodes = c(0, 0.25, 0.5, 1:5), approximation = "linear")
```

`curve.calibration` *Curve calibration*

Description

Function that calibrates and returns a Zero Coupon curve based on the coupon rates and IRR's of the assets. Uses the bootstrap method to find, recursively, the corresponding Zero Coupon rates given by the market data. This rates are then optimized by the minimization of the MAE between bond values given by the constructed rates and bond market value. Alternatively, uses minimization of residual sum of squares (RSS), allowing user to optimize or define an specific term structure of the segments.

Usage

```
curve.calibration(
  yield.curve,
  market.assets = NULL,
  noSpots = NULL,
  analysis.date = Sys.Date(),
  asset.type = "IBRSwaps",
  freq = 4,
  rate.type = 0,
  daycount = NULL,
  fwd = 0,
  npieces = NULL,
  obj = "Price",
  Weights = NULL,
  nsimul = 1,
  piece.term = NULL,
  nodes = seq(0, 10, 0.001),
  approximation = "linear"
)
```

Arguments

yield.curve	Internal rates of return of the market assets. Series of rates with different maturities that adjust to the time structure of the curve to calibrate and construct. names(yield.curve) needs to include the numeric maturities in years of each rate, unless the input market.assets includes the maturities as dates.
market.assets	Matrix containing the market assets. The first column contains the coupon rates of the assets and the second column represents their corresponding maturities as a date. This input is only required if IRR's of assets differ from coupon rate.
noSpots	Number of spot interest rates introduced in the yield.curve input. Function assumes spots are the first entries on the yield.curve vector.
analysis.date	Date for which the curve is going to be calibrated and, thus, constructed.
asset.type	String that determines the asset type to value. See also 'Details'.
freq	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
rate.type	(1) for annual compounded discount rates and (0) for continuosly compounded discount rates. By default, rates are assumed to be the former.
daycount	Day count convention. See also 'Details'.
fwd	Numeric value that determines if the desired output curve is a forward or a spot curve. Set 0 for spot curve (default), 1 otherwise.
npieces	Number of constant or linear segments for the curve to incorporate. By default NULL, and bootstrapping method is used, otherwise, minimization of RSS is used.
obj	String related to the definition of the error in the RSS methodology. Set "Price" for minimization of error by price or "Rate" for minimization of error by rate.
Weights	Vector of weights used to dot product with residual squares in order to calculate residual sum of squares. By default, each residual is assigned the same weight.
nsimul	Number of simulations for the terms of the pieces. The more simulations, the more likely to find a better local solution. By default 1, and terms are defined in such way each piece occupies the same length in the abscissa axis.
piece.term	Vector that establishes a unique term structure for optimization to take place. Each piece or segment must have a unique maturity, as numeric value in years, that signifies the end of the segment. Last segment maturity must not be introduced, it is assumed to be equivalent to the last term introduced on analysis date. Therefore, the piece.term vector must always have a length equal to npieces - 1.
nodes	Desired output nodes of the curve.
approximation	String that establish the approximation. Set 'linear' for a piecewise linear approximation, or 'constant' for a piecewise constant curve.

Details

asset.type makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "IBRSwaps" for swaps indexed to IBR rate.
- "LIBORSwaps" for Interest Rate Swaps (IRS) indexed to 3M LIBOR.
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.

If `npieces = NULL` uses a recursive iteration process based in bootstrapping where the curve is constructed through a minimization of the MAE between the dirty price of historical market assets and an approximation of the theoretical price of assets of same maturity. Uses the "L-BFGS-B" optimization method to minimize the expected MAE. Otherwise, curve is constructed through minimization of RSS where the error can be defined via price or rate.

Value

Zero Coupon curve for a specific date based on historical spot rates and bond structures.

Author(s)

Andres Galeano & Camilo Díaz

Examples

```
# Create input
yield.curve <- c(0.103,0.1034,0.1092, 0.1161, 0.1233, 0.1280, 0.1310, 0.1320, 0.1325, 0.1320)
names(yield.curve) <- c(0,0.08,0.25,0.5,1,2,3,5,7,10)
nodes <- seq(0,10,0.001)

market.assets <- matrix(NA,nrow = 10,ncol = 2)
market.assets[1,1] <- 0.1030 ; market.assets[2,1]  <- 0.1044
market.assets[3,1] <- 0.1083 ; market.assets[4,1]  <- 0.1010
market.assets[5,1] <- 0.1120 ; market.assets[6,1]  <- 0.1130
market.assets[7,1] <- 0.1150 ; market.assets[8,1]  <- 0.1160
market.assets[9,1] <- 0.1150 ; market.assets[10,1] <- 0.13
market.assets[1,2] <- "2019-01-03" ; market.assets[2,2]  <- "2019-02-03"
market.assets[3,2] <- "2019-04-03" ; market.assets[4,2]  <- "2019-07-03"
market.assets[5,2] <- "2020-01-03" ; market.assets[6,2]  <- "2021-01-03"
market.assets[7,2] <- "2022-01-03" ; market.assets[8,2]  <- "2024-07-03"
market.assets[9,2] <- "2026-01-03" ; market.assets[10,2] <- "2029-01-03"

# Function
curve.calibration (yield.curve = yield.curve, market.assets = market.assets,
                   analysis.date = "2019-01-03" , asset.type = "IBRSwaps",
                   freq = 4, daycount = "ACT/365", fwd = 0, nodes = nodes,
                   approximation = "linear")
```

`discount.factors` *Discount factors*

Description

Function that calculates discount factors given effective payment dates and a discount rate. Optional parameters available to calculate discrete or continuous discount factors.

Usage

```
discount.factors(
  dates,
  rates,
  analysis.date = Sys.Date(),
  rate.type = 1,
  freq = 1
)
```

Arguments

`dates`	Coupon payment dates.
`rates`	Discount rates given by the zero coupon rate curve. Can also be a unique discount rate.
`analysis.date`	Date in which the asset is valued. By default, the current date.
`rate.type`	(1) for discrete compounded discount rates and (0) for continuosly compounded discount rates. By default rates are assumed to be discrete.
`freq`	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).

Value

Discount factors.

Examples

```
discount.factors(dates = c("2020-09-10", "2020-12-10", "2021-03-10"), rates = c(0.07, 0.075, 0.08),
                 analysis.date = "2010-09-01")
discount.factors(dates = c("2025-09-01", "2025-12-01", "2026-03-01", "2026-06-01"),
                 rates = c(0.01, 0.015, 0.017, 0.02), analysis.date = "2025-06-01",
                 rate.type = 1, freq = 4)
```

discount.time *Quantil's discount time convention*

Description

Function to count the number of years between dates according to Quantil's discount convention. A year is defined as the difference in one year, between two dates with the exact month and day. Meanwhile, partial years are defined as the quotient between the number of elapsed days within a year and the total number of days that make up that year. Total number of years between the two dates is then the sum between complete full years and the partial portion.

Usage

```
discount.time(tinitial, tfinal)
```

Arguments

tinitial	Initial date of analysis.
tfinal	Final date of analysis.

Details

There is an exception. For example, for initial date 29-February, a year is defined as the 28 of February of the next year. Meanwhile four years, is defined as 29 of February four years after.

Value

Number of years between the specified dates.

Examples

```
discount.time(tinitial = "2024-07-13", tfinal = "2025-03-01")
discount.time(tinitial = "2024-02-29", tfinal = "2025-02-28")
discount.time(tinitial = "2024-02-29", tfinal = "2028-02-29")
```

fwd2spot *Forward curve conversion*

Description

Uses a recursive method to calculate the implicit spot rates of a given forward curve. Calculations and formulas based on the definition of forward rates where $\exp -rT = \exp - \int f(t)dt$.

Usage

```
fwd2spot(dates, fwd, approximation = "constant")
```

Arguments

dates	Term structure of rates.
fwd	Numeric vector of forward rates to be converted.
approximation	String that establish the approximation. Set 'linear' for a piecewise linear approximation, or 'constant' for a piecewise constant curve.

Details

Requires continuous rates. Recommended that the input forward curve starts with maturity 0, if not, function will approximate zero node as equal to node 1 (first term structure). Output forward curve slightly differs from empirical curve as it calculates an implicit forward curve.

Value

Implicit spot curve based on the input forward rates and input term structure.

Examples

```
# Inputs for calibration of forward curve
yield.curve <- c(0.015,0.0175, 0.0225, 0.0275, 0.0325, 0.0375,0.04,0.0425,0.045,0.0475,0.05)
names(yield.curve) <- c(0.5,1,2,3,4,5,6,7,8,9,10)
nodes <- seq(0,10,0.5)
# Calibration
fwd <- curve.calibration (yield.curve = yield.curve, market.assets = NULL,
                          analysis.date = "2019-01-03", asset.type = "IBRSwaps",
                          freq = 4, rate.type = 0, daycount = "ACT/365",  fwd = 1,
                          npieces = NULL, nodes = nodes, approximation = "constant")
# Forward to Spot
dates <- names(fwd)
fwd2spot(dates, fwd, approximation = "constant")
```

---

price.dirty2clean *From one price to another*

Description

Converts bond prices from dirty to clean and viceversa.

Usage

```
price.dirty2clean(
  maturity,
  analysis.date = Sys.Date(),
  price,
  dirty = 1,
  coupon.rate,
```

```
  principal = 1,
  asset.type = "TES",
  freq = NULL,
  daycount = "NL/365"
)
```

Arguments

`maturity`	Last day of the contract: YYYY-MM-DD. Alternatively, it can be a numeric value that represents the duration of the contract in years.
`analysis.date`	Date in which the asset is valued. By default, the current date.
`price`	Numeric value. Price of the bond to convert.
`dirty`	Numeric value. Determines if the input price corresponds to the dirty price or the clean price. For dirty price, set `dirty = 1`. Otherwise, `dirty = 0`
`coupon.rate`	Coupon rate of the asset. Can be an unique numeric value or a vector corresponding to each coupon payment date.
`principal`	Notional amount for the asset.
`asset.type`	String that determines the asset type to value. See also 'Details'.
`freq`	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
`daycount`	Day count convention. See also 'Details'.

Details

`asset.type` makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.
- "IBR" for bonds and assets indexed to 3M IBR rate.
- "LIBOR" for bonds and assets indexed to 3M LIBOR.

`daycount` convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.
- ACT/ACT-ISDA
- ACT/ACT-AFB

Value

The dirty price or clean price of a bond.

Examples

```
price.dirty2clean(maturity = "2026-01-03", analysis.date = "2023-01-02",
                  price = 1, dirty = 1, coupon.rate = 0.04, principal = 1)
price.dirty2clean(maturity = "2026-01-03", analysis.date = "2023-01-02",
                  price = 0.9601096, dirty = 0, coupon.rate = 0.04, principal = 1)
```

---

sens.bonds *Bond Sensitivity*

Description

Calculates the sensitivity of a given bond by numerically averaging the percentage change in bonds price when moving upwards and downwards, by 1 basic point, the Yield to Maturity vector.

Usage

```
sens.bonds(
  input,
  price,
  maturity,
  analysis.date = Sys.Date(),
  coupon.rate,
  principal = 1,
  asset.type = "TES",
  freq = 1,
  rate.type = 1,
  spread = 0,
  daycount = "ACT/365",
  dirty = 1,
  convention = "F",
  trade.date = NULL,
  coupon.schedule = "SF"
)
```

Arguments

`input`	String that establishes if the `price` input corresponds to the Internal Rate of Return (IRR) of the bond or the market price. Set `"rate"` for the IRR. Otherwise, `"price"`.
`price`	Numeric value of either market price or Internal Rate of Return of a given bond. Instead of IRR, can also be a vector of multiple rates, one for every coupon date.
`maturity`	Last day of the contract: YYYY-MM-DD. Alternatively, it can be a numeric value that represents the duration of the contract in years.
`analysis.date`	Date in which the asset is valued. By default, the current date.
`coupon.rate`	Coupon rate of the asset. Can be an unique numeric value or a vector corresponding to each coupon payment date.

`principal`	Notional amount for the asset.
`asset.type`	String that determines the asset type to value. See also 'Details'.
`freq`	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
`rate.type`	(1) for discrete compounded discount rates and (0) for continuosly compounded discount rates. By default rates are assumed to be discrete.
`spread`	Decimal value of spread added to coupon payment rate. By default, `0`.
`daycount`	Day count convention. See also 'Details'.
`dirty`	Numeric value to determine if the calculated price is dirty or clean. To calculate dirty price, set `dirty = 1`. Otherwise, `dirty = 0`.
`convention`	String that establishes if the effective dates are calculated using Following, Modified Following, Backward or Backward Following. See also 'Details'.
`trade.date`	The date on which the transaction occurs. It is used to calculate maturity as a date, when given in years. Also required for non-trivial cases such as bonds with long first coupon.
`coupon.schedule`	String that establishes if a bond first coupon period is a long first coupon or a short first coupon. On the contrary, establishes if last coupon period is long last coupon or a short last coupon. See also 'Details'.

Details

`asset.type` makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).
- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.
- "IBR" for bonds and assets indexed to 3M IBR rate.
- "LIBOR" for bonds and assets indexed to 3M LIBOR.

`daycount` convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.
- ACT/ACT-ISDA
- ACT/ACT-AFB

`convention` makes reference to the following type of business day conventions:

- "F" for Following business day convention.
- "MF" for Modified Following business day convention.

- "B" for Backward business day convention.
- "MB" for Modified Backward business day convention.

coupon.schedule makes reference to the following type of coupon payment schedule of a bond:

- "LF" for Long First coupon payment.
- "LL" for Long Last coupon payment.
- "SF" for Short First coupon payment.
- "SL" for Short Last coupon payment.

Value

Bond sensitivity

Examples

```
sens.bonds(input = c("price"), price = 0.98, maturity = "2023-01-03",
           analysis.date = "2019-01-05", coupon.rate = 0.04,
           principal = 1, asset.type = "IBR", rate.type = 1)
sens.bonds(input = c("rate"), price = rep(0.08,8), maturity = "2023-01-03",
           analysis.date = "2015-02-03", coupon.rate = 0.04,
           principal = 1, asset.type = "FixedIncome", freq = 1,
           rate.type = 1, daycount = "ACT/365", dirty = 1,
           convention = "MB", trade.date = "2015-02-03",
           coupon.schedule = "LF")
```

---

spot2forward *Spot curve conversion*

Description

Uses a recursive method to calculate the instantaneous forward rates of a given spot curve. Calculations and formulas based on the definition of forward rates where $\exp -rT = \exp - \int f(t)dt$.

Usage

```
spot2forward(dates, spot, approximation = "constant")
```

Arguments

dates	Term structure of rates.
spot	Vector of spot rates to be converted.
approximation	String that establish the approximation. Set 'linear' for a piecewise linear approximation, or 'constant' for a piecewise constant curve.

Details

Requires continuous rates. Recommended that the input spot curve starts with maturity 0, if not, input function will approximate zero node as equal to node 1 (first term structure). The time partition and available data affects calculation and precision of resulting forward curve. Output forward curve slightly differs from empirical curve as it calculates an implied instantaneous forward curve.

Value

Instantaneous forward curve based on the input spot and the input term structure.

Examples

```
# Inputs for calibration of spot curve
yield.curve <- c(0.015,0.0175, 0.0225, 0.0275, 0.0325, 0.0375,0.04,0.0425,0.045,0.0475,0.05)
names(yield.curve) <- c(0.5,1,2,3,4,5,6,7,8,9,10)
nodes <- seq(0,10,0.001)
# Calibration
spot <- curve.calibration (yield.curve = yield.curve, market.assets = NULL,
                           analysis.date = "2019-01-03", asset.type = "IBRSwaps",
                           freq = 4, rate.type = 0, fwd = 0, npieces = NULL,
                           obj = "Price", nodes = nodes, approximation = "linear")
# Spot to Forward
dates <- names(spot)
spot2forward(dates, spot, approximation = "linear")
```

---

valuation.bonds *Bond valuation*

Description

Function that values various asset types with varying payment frequencies. It covers fixed-coupon assets, spread income assets, floating notes and fixed legs of interest rate swaps.

Usage

```
valuation.bonds(
  maturity,
  coupon.rate,
  rates,
  principal = 1,
  analysis.date = Sys.Date(),
  asset.type = "TES",
  freq = NULL,
  rate.type = 1,
  spread = 0,
  daycount = "NL/365",
  dirty = 1,
```

```
  convention = "F",
  trade.date = NULL,
  coupon.schedule = "SF",
  spread.only = FALSE
)
```

Arguments

maturity	Last day of the contract: YYYY-MM-DD. Alternatively, it can be a numeric value that represents the duration of the contract in years.
coupon.rate	Coupon rate of the asset. Can be an unique numeric value or a vector corresponding to each coupon payment date.
rates	Discount rates given by the zero coupon rate curve. Can also be a unique discount rate.
principal	Notional amount for the asset.
analysis.date	Date in which the asset is valued. By default, the current date.
asset.type	String that determines the asset type to value. See also 'Details'.
freq	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
rate.type	(1) for annual compounded discount rates and (0) for continuosly compounded discount rates. By default, rates are assumed to be the former.
spread	Decimal value of spread added to coupon payment rate. By default, 0.
daycount	Day count convention. See also 'Details'.
dirty	Numeric value to determine if the calculated price is dirty or clean. To calculate dirty price, set dirty = 1. Otherwise, dirty = 0.
convention	String that establishes if the effective dates are calculated using Following, Modified Following, Backward or Backward Following. See also 'Details'.
trade.date	The date on which the transaction occurs. It is used to calculate maturity as a date, when given in years. Also required for non-trivial cases such as bonds with long first coupon.
coupon.schedule	String that establishes if a bond first coupon period is a long first coupon or a short first coupon. On the contrary, establishes if last coupon period is long last coupon or a short last coupon. See also 'Details'.
spread.only	Logical condition that establishes if the output should just include the spread or the complete bond value. By default, FALSE, referring to the output being bond value.

Details

asset.type makes reference to the following type of assets:

- "TES" for Colombian Treasury Bonds (default).

- "FixedIncome" for assets that are indexed to a fixed income with different frequency of payments.
- "IBR" for bonds and assets indexed to 3M IBR rate.
- "LIBOR" for bonds and assets indexed to 3M LIBOR.

`daycount` convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.
- ACT/ACT-ISDA
- ACT/ACT-AFB

`convention` makes reference to the following type of business day conventions:

- "F" for Following business day convention.
- "MF" for Modified Following business day convention.
- "B" for Backward business day convention.
- "MB" for Modified Backward business day convention.

`coupon.schedule` makes reference to the following type of coupon payment schedule of a bond:

- "LF" for Long First coupon payment.
- "LL" for Long Last coupon payment.
- "SF" for Short First coupon payment.
- "SL" for Short Last coupon payment.

Value

Bond value.

Examples

```
valuation.bonds(maturity = "2026-06-01", coupon.rate = 0.06, rates = 0.08,
                analysis.date = "2022-06-01")
valuation.bonds(maturity = "2026-06-01", coupon.rate = 0.06, rates = rep(0.08,4),
                analysis.date = "2022-06-01", rate.type = 0)
valuation.bonds(maturity = "2026-06-01", analysis.date= "2025-02-27",
                coupon.rate = c(0.06, 0.062, 0.063, 0.065, 0.066, 0.068),
                rates = c(0.08, 0.082, 0.078, 0.09, 0.077, 0.085),
                asset.type = "IBR")
valuation.bonds(maturity = "2026-06-01", coupon.rate = 0.06,
                rates = 0.08, asset.type = "IBR", freq = 4,
                spread = 0.03)
valuation.bonds(maturity = 4.58, coupon.rate = 0.1256, rates = seq(0.05, 0.14, by = 0.005),
                analysis.date = "2019-07-14", asset.type = "FixedIncome", freq = 4,
           principal = 567, daycount = "ACT/360", rate.type = 0, trade.date = "2019-07-14",
                coupon.schedule = "LL")
```

`valuation.swaps` *Swap valuation*

Description

Function that values Interest Rate Swaps (IRS) and Cross Currency Swaps (CCS).

Usage

```
valuation.swaps(
  maturity,
  analysis.date = Sys.Date(),
  asset.type = "IBRSwaps",
  freq = 4,
  coupon.rate,
  rates,
  float.rate = NULL,
  spread = 0,
  principal = 1,
  Legs = "FF",
  ex.rate = NULL,
  basis.rates = NULL,
  coupon.rate2 = NULL,
  rates2 = NULL,
  float.rate2 = NULL,
  spread2 = 0,
  principal2 = NULL,
  rate.type = 1,
  daycount = "NL/365",
  loc = "BOG",
  convention = "F",
  trade.date = NULL,
  coupon.schedule = "SF"
)
```

Arguments

`maturity`	Last day of the contract: YYYY-MM-DD. Alternatively, it can be a numeric value that represents the duration of the contract in years.
`analysis.date`	Date in which the asset is valued. By default, the current date.
`asset.type`	String that determines the asset type to value. See also 'Details'.
`freq`	Frequency of payments of a given asset in a year. For LIBOR and IBR the default frequency is four (quarterly payments). TES has a default frequency of one (annual payments).
`coupon.rate`	For (IRS), coupon rate of the fixed leg. For (CCS), coupon rate of local fixed leg.

rates	For (IRS) discount rates given by the zero coupon rate curve. For (CCS), represents discount rates for local currency. Can be a vector that corresponds to each coupon date or a curve with at least, nodes with 3 decimals.
float.rate	For (IRS), last observed floating rate, necessary for variable leg when swap valuation date doesn't belong to a coupon date. For (CCS), last observed local floating rate. By default, NULL.
spread	Decimal value of spread added to coupon payment rate. By default, 0.
principal	For (IRS), notional amount for both legs. For (CCS), notional amount of local leg.
Legs	For (CCS), string that establishes the type of both legs that makeup the Cross Currency Swap. See also 'Details'.
ex.rate	Exchange rate on analysis date. Format has to be local currency divided by foreign currency.
basis.rates	"Discount rates" given by the basis curve. Can be a vector that corresponds to each coupon date or a curve with at least, nodes with 3 decimals.
coupon.rate2	Coupon rate of the foreign leg. By default, NULL.
rates2	Discount rates given by the foreign zero coupon rate curve. Can be a vector that corresponds to each coupon date or a curve with at least, nodes with 3 decimals.
float.rate2	Last observed foreign floating rate. By default, NULL.
spread2	Decimal value of spread added to foreign floating rate. By default, 0.
principal2	Notional amount for the foreign leg. By default, NULL.
rate.type	(1) for annual compounded discount rates and (0) for continuosly compounded discount rates. By default, rates are assumed to be the former.
daycount	Day count convention. See also 'Details'.
loc	String related to the location of the asset. It is used to calculate the effective dates, taking into account the business days of the given location. See also 'Details'.
convention	String that establishes if the effective dates are calculated using Following, Modified Following, Backward or Backward Following. See also 'Details'.
trade.date	The date on which the transaction occurs. It is used to calculate maturity as a date, when given in years. Also required for non-trivial cases such as bonds with long first coupon.
coupon.schedule	String that establishes if a bond first coupon period is a long first coupon or a short first coupon. On the contrary, establishes if last coupon period is long last coupon or a short last coupon. See also 'Details'.

Details

asset.type makes reference to the following type of assets:

- "IBRSwaps" for swaps indexed to IBR rate.
- "LIBORSwaps" for Interest Rate Swaps (IRS) indexed to 3M LIBOR.

- "CCS" for cross currency swaps.

`daycount` convention accepts the following values:

- 30/360.
- ACT/365.
- ACT/360 (Default).
- ACT/365L.
- NL/365.
- ACT/ACT-ISDA
- ACT/ACT-AFB

`convention` makes reference to the following type of business day conventions:

- "F" for Following business day convention.
- "MF" for Modified Following business day convention.
- "B" for Backward business day convention.
- "MB" for Modified Backward business day convention.

`coupon.schedule` makes reference to the following type of coupon payment schedule of a bond:

- "LF" for Long First coupon payment.
- "LL" for Long Last coupon payment.
- "SF" for Short First coupon payment.
- "SL" for Short Last coupon payment.

`Legs` makes reference to the following types of legs composition of the cross currency swap

- "FF" for fixed leg in local currency and fixed leg in foreign currency.
- "FV" for fixed leg in local currency and variable leg in foreign currency.
- "VF" for variable leg in local currency and fixed leg in foreign currency.
- "VV" for variable leg in local currency and variable leg in foreign currency.

Value

Swap value for Interest Rate Swap (IRS) or Cross Currency Swap (CCR).

Examples

```
# (IRS) ------------------------------------------------------------------------
valuation.swaps(maturity = "2026-06-01", analysis.date= Sys.Date(),
                coupon.rate = 0.06, rates = 0.08, float.rate = 0.03)
valuation.swaps(maturity = "2021-03-09", analysis.date= "2018-03-09",
                coupon.rate = 0.05, rates = 0.08, rate.type = 0)
valuation.swaps(maturity = "2026-07-01", analysis.date = "2023-01-02",
                asset.type = "IBRSwaps", freq = 4,
                coupon.rate = 0.04, rates = rep(0.05,14),
                float.rate = 0.03, spread = 0)
```

```
# (CCS) ------------------------------------------------------------------
# Curve Calibration for rates input
yield.curve <- c(0.103,0.1034,0.1092, 0.1161, 0.1233, 0.1280, 0.1310, 0.1320, 0.1325)
names(yield.curve) <- c(0,0.08,0.25,0.5,1,2,3,5,6)
nodes <- seq(0, 10, by = 0.001) # Our curve has nodes with three decimals.
rates <- curve.calibration (yield.curve = yield.curve, market.assets = NULL,
                            analysis.date = "2023-03-01", asset.type = "IBRSwaps",
                            freq = 4, rate.type = 0, daycount= "ACT/365",
                       fwd = 0, npieces = NULL, nodes = nodes, approximation = "constant")

# Curve Calibration for basis.rates input
nodes  <- seq(0, 10, by = 0.001)
rates2 <- rates/4 # It is assumed foreign curve is proportional to local spot curve.
# Swaps input for calibration
ex.rate <- 4814
swaps <- rbind(c("2024-03-01", "FF", 0.07 , 0.0325, NA   , NA    , 2000 * ex.rate, 2000),
c("2025-03-01", "VV", NA   , NA    , 0.015, 0.0175, 2000 * ex.rate, 2000),
c("2026-03-01", "FF", 0.075, 0.03  , NA   ,  NA   , 500000, 5000000 / ex.rate),
c("2027-03-01", "VV", NA   , NA    , 0.01 , 0.015 , 5000000, 5000000 / ex.rate),
c("2028-03-01", "FF", 0.08 ,0.035  , NA   , NA    , 3000000, 3000000 / ex.rate),
c("2029-03-01", "VV", NA   , NA    , 0.01 , 0.0125, 3000000, 3000000 / ex.rate))
colnames(swaps) <- c("Mat"  ,"Legs", "C1" , "C2", "spread1", "spread2", "prin1", "prin2")
# Calibration
basis.rates <- basis.curve(swaps, ex.rate = 4814, analysis.date = "2023-03-01",
                           freq = c(2,2,2,2,1,1), rates = rates, rates2 = rates2,
                           rate.type = 1, npieces = NULL, obj = "Price",
                           Weights = NULL, nodes = nodes, approximation = "linear")

# Valuation
valuation.swaps (maturity = "2024-03-01", analysis.date = "2023-03-01",
                 asset.type = "CCS", freq = 2, Legs = "FF", ex.rate = 4814,
                 coupon.rate = 0.07, coupon.rate2 = 0.0325,
                 rates = rates, rates2 = rates2, basis.rates = basis.rates,
                 float.rate = NULL, float.rate2 = NULL, spread = 0,
                 spread2 = 0, principal = 2000 * 4814, principal2 = 2000,
                 rate.type = 0, daycount = "ACT/365", loc = "BOG",
                 convention = "F")
```

Index