Package 'brfinance'

February 19, 2026

Title Access to Brazilian Macroeconomic and Financial Time Series

Version 0.8.0

Description Provides simplified access to selected Brazilian macroeconomic and financial time series from official sources, primarily the Central Bank of Brazil through the SGS (Sistema Gerenciador de Séries Temporais) API. The package enables users to quickly retrieve and visualize indicators such as the unemployment rate and the Selic interest rate using a standardized data structure. It is designed for data access and visualization purposes, without performing forecasts or statistical modeling. For more information, see the official API: <https://dadosabertos.bcb.gov.br/dataset/>.

License MIT + file LICENSE

URL https://github.com/efram2/brfinance

BugReports https://github.com/efram2/brfinance/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports dplyr, ggplot2, scales, httr2, lubridate, labelled,

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

VignetteBuilder knitr

NeedsCompilation no

Author João Paulo dos Santos Pereira Barbosa [aut, cre]

Maintainer João Paulo dos Santos Pereira Barbosa <joao.31582129@gmail.com>

Repository CRAN

Date/Publication 2026-02-19 08:30:27 UTC

Contents

browse_series 2
br_available_series 3

calc_amortization_schedule . . . 5
calc_compound_interest . . . 5
calc_continuous_compounding . . . 6
calc_effective_rate . . . 7
calc_future_value . . . 7
calc_future_value_ext . . . 8
calc_fv_annuity . . . 9
calc_irr . . . 9
calc_nominal_rate . . . 10
calc_nper . . . 10
calc_npv . . . 11
calc_pmt . . . 12
calc_present_value . . . 12
calc_pv_annuity . . . 13
calc_pv_continuous . . . 14
calc_rate . . . 14
calc_simple_interest . . . 15
get_cdi_rate . . . 16
get_exchange_rate . . . 17
get_gdp_growth . . . 19
get_inflation_rate . . . 20
get_selic_rate . . . 22
get_series_info . . . 24
get_unemployment . . . 24
plot_cdi_rate . . . 26
plot_exchange_rate . . . 26
plot_inflation_rate . . . 27
plot_selic_rate . . . 28
plot_series_comparison . . . 29
plot_unemployment . . . 30
rule_of_114 . . . 31
rule_of_72 . . . 31

Index **33**

`browse_series` *Browse Available Economic Series*

Description

Displays a searchable table of available Brazilian economic series.

Usage

```
browse_series(
  category = NULL,
  sub_category = NULL,
  source = NULL,
  frequency = NULL,
  search_text = NULL,
  language = c("pt", "en")
)
```

Arguments

category	Filter by category (optional)
sub_category	Filter by sub-category (optional)
source	Filter by source (optional)
frequency	Filter by frequency (optional)
search_text	Text search across all columns (optional)
language	Language for returned labels. Either "pt" or "en".

Value

A tibble with available series matching the criteria

Examples

```
browse_series()
browse_series(category = "Macroeconomic")
browse_series(search_text = "inflação")
```

---

br_available_series *Brazilian Economic Series Database*

Description

A comprehensive database of Brazilian economic and financial time series available through the Central Bank of Brazil's SGS system.

Usage

```
br_available_series
```

Format

A tibble with multiple rows and the following columns:

series_id SGS code from Central Bank of Brazil

source Data source

source_system Source system

short_name Short name of the series

long_name_pt Full name in Portuguese

long_name_en Full name in English

description_pt Description in Portuguese

description_en Description in English

category_pt Category in Portuguese

category_en Category in English

sub_category_pt Sub-category in Portuguese

sub_category_en Sub-category in English

frequency Data frequency

unit Measurement unit

seasonally_adjusted Logical

start_date Series start date

end_date Series end date

default_transform Suggested transformation

suggested_scale Suggested visualization scale

notes_pt Notes in Portuguese

notes_en Notes in English

Source

Central Bank of Brazil — SGS

Examples

```
br_available_series

br_available_series[br_available_series$series_id == "433", ]

br_available_series[br_available_series$series_id == "433", ]
```

calc_amortization_schedule

Amortization Schedule

Description

Generates an amortization schedule for a loan.

Usage

```
calc_amortization_schedule(pv, rate, n, type = 0)
```

Arguments

pv	Loan amount.
rate	Interest rate per period (decimal).
n	Number of periods.
type	Payment timing: 0 for end of period, 1 for beginning of period.

Value

Data frame with amortization schedule.

Examples

```
calc_amortization_schedule(pv = 10000, rate = 0.01, n = 12)
```

calc_compound_interest

Compound Interest

Description

Calculates the final value under compound interest.

Usage

```
calc_compound_interest(pv, rate, n)
```

Arguments

pv	Present value.
rate	Interest rate per period (decimal).
n	Number of periods.

Value

Final value with compound interest.

Examples

```
calc_compound_interest(pv = 1000, rate = 0.10, n = 2)
```

`calc_continuous_compounding` *Continuous Compounding*

Description

Calculates the future value with continuous compounding.

Usage

```
calc_continuous_compounding(pv, rate, t)
```

Arguments

`pv`	Present value.
`rate`	Nominal annual interest rate (decimal).
`t`	Time in years.

Value

Future value with continuous compounding.

Examples

```
calc_continuous_compounding(pv = 1000, rate = 0.05, t = 2)
```

calc_effective_rate *Effective Interest Rate*

Description

Converts a nominal interest rate into an effective rate.

Usage

```
calc_effective_rate(nominal_rate, m)
```

Arguments

nominal_rate	Nominal interest rate (decimal).
m	Number of compounding periods per year.

Value

Effective interest rate.

Examples

```
calc_effective_rate(nominal_rate = 0.12, m = 12) # 12% nominal monthly
```

calc_future_value *Future Value (FV) - Basic*

Description

Calculates the future value of a present amount given an interest rate and number of periods.

Usage

```
calc_future_value(pv, rate, n)
```

Arguments

pv	Present value.
rate	Interest rate per period (decimal).
n	Number of periods.

Value

Future value.

Examples

```
calc_future_value(pv = 1000, rate = 0.10, n = 1)
calc_future_value(pv = 500, rate = 0.05, n = 10)
```

`calc_future_value_ext` *Future Value Extended (with periodic payments)*

Description

Calculates the future value with optional periodic payments.

Usage

```
calc_future_value_ext(pv, rate, n, pmt = 0, type = 0)
```

Arguments

`pv`	Present value.
`rate`	Interest rate per period (decimal).
`n`	Number of periods.
`pmt`	Periodic payment (default = 0).
`type`	Payment timing: 0 for end of period (ordinary annuity), 1 for beginning of period (annuity due).

Value

Future value.

Examples

```
calc_future_value_ext(pv = 1000, rate = 0.05, n = 10, pmt = 100)
calc_future_value_ext(pv = 0, rate = 0.01, n = 12, pmt = 500, type = 1)
```

calc_fv_annuity *Future Value of an Annuity*

Description

Calculates the future value of a series of equal payments.

Usage

```
calc_fv_annuity(pmt, rate, n, type = 0)
```

Arguments

pmt	Payment amount per period.
rate	Interest rate per period (decimal).
n	Number of periods.
type	Payment timing: 0 for end of period (ordinary annuity), 1 for beginning of period (annuity due).

Value

Future value of the annuity.

Examples

```
calc_fv_annuity(pmt = 100, rate = 0.05, n = 10, type = 0)
calc_fv_annuity(pmt = 100, rate = 0.05, n = 10, type = 1)
```

calc_irr *Internal Rate of Return (IRR)*

Description

Calculates the internal rate of return of a series of cash flows.

Usage

```
calc_irr(cashflows, guess = 0.1)
```

Arguments

cashflows	Vector of cash flows (first is typically negative investment).
guess	Initial guess for IRR (default = 0.1).

Value

Internal rate of return.

Examples

```
calc_irr(cashflows = c(-1000, 300, 400, 500))
```

---

calc_nominal_rate *Nominal Interest Rate*

Description

Converts an effective interest rate into a nominal rate.

Usage

```
calc_nominal_rate(effective_rate, m)
```

Arguments

effective_rate	Effective interest rate (decimal).
m	Number of compounding periods per year.

Value

Nominal interest rate.

Examples

```
calc_nominal_rate(effective_rate = 0.1268, m = 12)
```

---

calc_nper *Number of Periods (NPER)*

Description

Calculates the number of periods for an investment.

Usage

```
calc_nper(pv, fv = 0, rate, pmt = 0, type = 0)
```

Arguments

pv	Present value.
fv	Future value.
rate	Interest rate per period (decimal).
pmt	Payment per period (optional, default = 0).
type	Payment timing: 0 for end of period, 1 for beginning of period.

Value

Number of periods.

Examples

```
calc_nper(pv = -1000, fv = 2000, rate = 0.05)
calc_nper(pv = 0, fv = 100000, rate = 0.01, pmt = -500)
```

---

calc_npv *Net Present Value (NPV)*

Description

Calculates the net present value of a series of cash flows.

Usage

```
calc_npv(rate, cashflows)
```

Arguments

rate	Discount rate per period (decimal).
cashflows	Vector of cash flows (first is typically negative investment).

Value

Net present value.

Examples

```
calc_npv(rate = 0.1, cashflows = c(-1000, 300, 400, 500))
```

calc_pmt *Loan Payment (PMT)*

Description

Calculates the periodic payment for a loan.

Usage

```
calc_pmt(pv, rate, n, type = 0)
```

Arguments

pv	Present value (loan amount).
rate	Interest rate per period (decimal).
n	Number of periods.
type	Payment timing: 0 for end of period, 1 for beginning of period.

Value

Periodic payment amount.

Examples

```
calc_pmt(pv = 10000, rate = 0.01, n = 12) # Loan de R$10k, 1% ao mês, 12 meses
```

calc_present_value *Present Value (PV)*

Description

Calculates the present value of a future amount given an interest rate and number of periods.

Usage

```
calc_present_value(fv, rate, n)
```

Arguments

fv	Future value.
rate	Interest rate per period (decimal, e.g. 0.05 for 5%).
n	Number of periods.

Value

Present value.

Examples

```
calc_present_value(fv = 1100, rate = 0.10, n = 1)
calc_present_value(fv = 1000, rate = 0.05, n = 5)
```

---

calc_pv_annuity *Present Value of an Annuity*

Description

Calculates the present value of a series of equal payments.

Usage

```
calc_pv_annuity(pmt, rate, n, type = 0)
```

Arguments

pmt	Payment amount per period.
rate	Interest rate per period (decimal).
n	Number of periods.
type	Payment timing: 0 for end of period, 1 for beginning of period.

Value

Present value of the annuity.

Examples

```
calc_pv_annuity(pmt = 100, rate = 0.05, n = 10)
calc_pv_annuity(pmt = 100, rate = 0.05, n = 10, type = 1)
```

calc_pv_continuous *Present Value with Continuous Compounding*

Description

Calculates the present value with continuous compounding.

Usage

```
calc_pv_continuous(fv, rate, t)
```

Arguments

fv	Future value.
rate	Nominal annual interest rate (decimal).
t	Time in years.

Value

Present value with continuous compounding.

Examples

```
calc_pv_continuous(fv = 1105.17, rate = 0.05, t = 2)
```

calc_rate *Interest Rate (RATE)*

Description

Calculates the interest rate per period.

Usage

```
calc_rate(
  n,
  pv,
  fv = 0,
  pmt = 0,
  type = 0,
  guess = 0.1,
  max_iter = 100,
  tol = 1e-08
)
```

Arguments

n	Number of periods.
pv	Present value.
fv	Future value.
pmt	Payment per period (optional, default = 0).
type	Payment timing: 0 for end of period, 1 for beginning of period.
guess	Initial guess for the rate (default = 0.1).
max_iter	Maximum number of iterations (default = 100).
tol	Tolerance for convergence (default = 1e-8).

Value

Interest rate per period.

Examples

```
calc_rate(n = 12, pv = -1000, fv = 2000)
calc_rate(n = 60, pv = -20000, fv = 0, pmt = 386.66)
```

---

calc_simple_interest *Simple Interest*

Description

Calculates the final value under simple interest.

Usage

```
calc_simple_interest(pv, rate, n)
```

Arguments

pv	Present value.
rate	Interest rate per period (decimal).
n	Number of periods.

Value

Final value with simple interest.

Examples

```
calc_simple_interest(pv = 1000, rate = 0.10, n = 2)
```

get_cdi_rate *Get CDI Rate (Interbank Deposit Certificate)*

Description

Downloads daily CDI (Certificado de Depósito Interbancário) rate from BCB/SGS. This function retrieves the daily CDI rate (SGS series 12), which is the benchmark interest rate for interbank transactions in Brazil.

Usage

```
get_cdi_rate(
  start_date = NULL,
  end_date = NULL,
  language = "eng",
  labels = TRUE
)
```

Arguments

start_date	Start date for the data period. Accepts multiple formats: • "YYYY" for year only (e.g., "2020" becomes "2020-01-01") • "YYYY-MM" for year and month (e.g., "2020-06" becomes "2020-06-01") • "YYYY-MM-DD" for a specific date (e.g., "2020-06-15")
end_date	End date for the data period. Accepts the same formats as start_date: • "YYYY" (e.g., "2023" becomes "2023-12-31") • "YYYY-MM" (e.g., "2023-12" becomes the last day of December 2023) • "YYYY-MM-DD" for a specific date • NULL defaults to the current date (today)
language	Language for column names in the returned data.frame: • "eng" (default): Returns columns date and cdi_rate • "pt": Returns columns data_referencia and taxa_cdi
labels	Logical indicating whether to add variable labels using the labelled package. Labels provide descriptive text for each column when available.

Value

A data.frame with columns:

date Reference date

value Daily CDI rate (% per day)

value_annualized Annualized CDI rate (% per year, 252 business days)

Note

Series information: This function uses SGS series 12, which represents the daily CDI rate (taxa de juros - CDI). The CDI is a key benchmark for fixed income investments and interbank lending in Brazil. Data is available from 1986 onward with daily frequency (business days only).

Examples

```
# Default: last 30 days of CDI rate
df <- get_cdi_rate()

# Specific period
df2 <- get_cdi_rate("2023-01-01", "2023-03-31")

# Using year-month format for a specific month
df3 <- get_cdi_rate("2023-06", "2023-06")

# Portuguese column names and labels
df4 <- get_cdi_rate(language = "pt")

# Complete example with all parameters
df5 <- get_cdi_rate("2023-01-01", "2023-12-31", language = "pt", labels = TRUE)

# Historical analysis
df6 <- get_cdi_rate("2020-03-01", "2020-04-30")  # COVID period
```

---

get_exchange_rate *Get US Dollar Exchange Rate (Commercial)*

Description

Downloads daily US dollar exchange rate (commercial, selling) from BCB/SGS. This function retrieves the daily exchange rate (SGS series 1) in Brazilian Real (R$).

Usage

```
get_exchange_rate(
  start_date = NULL,
  end_date = NULL,
  language = "eng",
  labels = TRUE
)
```

Arguments

`start_date` Start date for the data period. Accepts multiple formats:

- `"YYYY"` for year only (e.g., `"2020"` becomes `"2020-01-01"`)
- `"YYYY-MM"` for year and month (e.g., `"2020-06"` becomes `"2020-06-01"`)

- "YYYY-MM-DD" for a specific date (e.g., "2020-06-15")

end_date — End date for the data period. Accepts the same formats as start_date:

- "YYYY" (e.g., "2023" becomes "2023-12-31")
- "YYYY-MM" (e.g., "2023-12" becomes the last day of December 2023)
- "YYYY-MM-DD" for a specific date
- NULL defaults to the current date (today)

language — Language for column names in the returned data.frame:

- "eng" (default): Returns columns date and exchange_rate
- "pt": Returns columns data_referencia and taxa_cambio

labels — Logical indicating whether to add variable labels using the labelled package. Labels provide descriptive text for each column when available.

Value

A data.frame with US dollar exchange rate. Columns depend on the language parameter:

- English (language = "eng"): date (Date), exchange_rate (numeric, R$/US$)
- Portuguese (language = "pt"): data_referencia (Date), taxa_cambio (numeric, R$/US$)

Note

Series information: This function uses SGS series 1, which represents the commercial US dollar selling rate (taxa de câmbio livre - dólar americano - venda). Data is available from 1984 onward with daily frequency.

Examples

```
# Default: last 30 days of exchange rate
df <- get_exchange_rate()

# Specific period
df2 <- get_exchange_rate("2023-01-01", "2023-03-31")

# Using year-month format for a specific month
df3 <- get_exchange_rate("2023-06", "2023-06")

# Portuguese column names and labels
df4 <- get_exchange_rate(language = "pt")

# Complete example with all parameters
df5 <- get_exchange_rate("2023-01-01", "2023-12-31", language = "pt", labels = TRUE)
```

get_gdp_growth *Get GDP Growth Rate*

Description

Downloads quarterly GDP growth data (% change) from BCB/SGS (Brazilian Central Bank). This function retrieves nominal GDP values (SGS series 2010) and calculates the quarter-over-quarter growth rate.

Usage

```
get_gdp_growth(
  start_date = "2000-01-01",
  end_date = NULL,
  language = "eng",
  labels = TRUE
)
```

Arguments

start_date	Start date for the data period. Accepts multiple formats: • "YYYY" for year only (e.g., "2020" becomes "2020-01-01") • "YYYY-MM" for year and month (e.g., "2020-06" becomes "2020-06-01") • "YYYY-MM-DD" for a specific date (e.g., "2020-06-15")
end_date	End date for the data period. Accepts the same formats as start_date: • "YYYY" (e.g., "2023" becomes "2023-12-31") • "YYYY-MM" (e.g., "2023-12" becomes the last day of December 2023) • "YYYY-MM-DD" for a specific date • NULL defaults to the current date (today)
language	Language for column names in the returned data.frame: • "eng" (default): Returns columns date and gdp_growth • "pt": Returns columns data_referencia and crescimento_pib
labels	Logical indicating whether to add variable labels using the labelled package. Labels provide descriptive text for each column when available.

Value

A data.frame with GDP growth rate. Columns depend on the language parameter:

- English (language = "eng"): date (Date), gdp_growth (numeric, %)
- Portuguese (language = "pt"): data_referencia (Date), crescimento_pib (numeric, %)

Note

Important limitation: The nominal GDP series (SGS 2010) is currently available only until 2014. Requests for periods after 2014 will return empty results or a warning.

Examples

```
# Default: data from 2000 to current date (but limited to 2014)
df <- get_gdp_growth()

# Specific period (within available range)
df2 <- get_gdp_growth("2010", "2014")

# Using year-month format
df3 <- get_gdp_growth("2012-06", "2013-12")

# End date only (from earliest available to 2020-12-31)
df4 <- get_gdp_growth(end_date = "2020-12-01")

# Portuguese column names and labels
df5 <- get_gdp_growth(language = "pt")

# Complete example with all parameters
df6 <- get_gdp_growth("2011-01-01", "2014-12-31", language = "pt", labels = TRUE)
```

---

get_inflation_rate *Get IPCA Inflation Data*

Description

Downloads monthly IPCA (Broad National Consumer Price Index) inflation data from the Brazilian Central Bank's SGS API and calculates accumulated inflation rates. IPCA is Brazil's official inflation index, calculated monthly by IBGE.

Usage

```
get_inflation_rate(
  start_date = "2012-01-01",
  end_date = NULL,
  language = "eng",
  labels = TRUE
)
```

Arguments

start_date	Start date for the data period. Accepts multiple formats: • "YYYY" for year only (e.g., "2020" becomes "2020-01-01") • "YYYY-MM" for year and month (e.g., "2020-06" becomes "2020-06-01") • "YYYY-MM-DD" for a specific date (e.g., "2020-06-15") • NULL defaults to "2020-01-01" (aligned with other functions in the package)
end_date	End date for the data period. Accepts the same formats as start_date:

- "YYYY" (e.g., "2023" becomes "2023-12-31")
- "YYYY-MM" (e.g., "2023-12" becomes the last day of December 2023)
- "YYYY-MM-DD" for a specific date
- NULL defaults to the current date (today)

language — Language for column names in the returned data.frame:

- "eng" (default): Returns columns date, monthly_inflation, ytd_inflation, twelve_month_inflation
- "pt": Returns columns data_referencia, inflacao_mensal, inflacao_acumulada_ano, inflacao_12_meses

labels — Logical indicating whether to add variable labels using the labelled package. Labels provide descriptive text for each column when available.

Value

A data.frame with inflation metrics. Columns depend on the language parameter:

- English (language = "eng"):
  - date (Date): Reference month
  - monthly_inflation (numeric): Monthly IPCA variation (%)
  - ytd_inflation (numeric): Year-to-date accumulated inflation (%)
  - twelve_month_inflation (numeric): 12-month accumulated inflation (%)
- Portuguese (language = "pt"):
  - data_referencia (Date): Mes de referencia
  - inflacao_mensal (numeric): Variacao mensal do IPCA (%)
  - inflacao_acumulada_ano (numeric): Inflacao acumulada no ano (%)
  - inflacao_12_meses (numeric): Inflacao acumulada nos ultimos 12 meses (%)

Note

Default Period: When start_date = NULL, defaults to "2020-01-01", providing data from the start of 2020. This period covers significant economic events including the COVID-19 pandemic and recent inflationary pressures in Brazil.

Data Processing: This function automatically downloads an extra 12 months of historical data to calculate 12-month accumulated inflation correctly. The final output is filtered to show only the requested period.

Calculation Details:

- Year-to-date inflation: Cumulative product of monthly rates within each calendar year
- 12-month inflation: Rolling 12-month cumulative product of monthly rates

Examples

```
# Default: from 2020 to current date (aligned with SELIC function)
df <- get_inflation_rate()

# Specific period with year-only format
```

```
df2 <- get_inflation_rate("2021", "2023")

# Using year-month format for precise month selection
df3 <- get_inflation_rate("2022-03", "2023-06")

# Portuguese column names and labels
df4 <- get_inflation_rate(language = "pt")

# Without variable labels
df5 <- get_inflation_rate("2020-01-01", "2022-12-31", labels = FALSE)

# Current year analysis
current_year <- format(Sys.Date(), "%Y")
df6 <- get_inflation_rate(start_date = current_year)

# Compare with SELIC rate (same default period)
selic_data <- get_selic_rate()  # Also starts at 2020-01-01
inflation_data <- get_inflation_rate()  # Same start date
```

`get_selic_rate` *Get Annual Brazilian SELIC Rate (Annualized, Base 252)*

Description

Downloads the annual SELIC rate series from the Central Bank of Brazil's SGS API. The SELIC rate (Special System for Settlement and Custody) is Brazil's benchmark overnight interest rate, used as the primary monetary policy instrument.

Usage

```
get_selic_rate(
  start_date = "2020-01-01",
  end_date = NULL,
  language = "eng",
  labels = TRUE
)
```

Arguments

`start_date`	Start date for the data period. Accepts multiple formats: • `"YYYY"` for year only (e.g., `"2020"` becomes `"2020-01-01"`) • `"YYYY-MM"` for year and month (e.g., `"2020-06"` becomes `"2020-06-01"`) • `"YYYY-MM-DD"` for a specific date (e.g., `"2020-06-15"`) • `NULL` defaults to `"2020-01-01"` (sensible default for analysis)
`end_date`	End date for the data period. Accepts the same formats as `start_date`: • `"YYYY"` (e.g., `"2023"` becomes `"2023-12-31"`)

- "YYYY-MM" (e.g., "2023-12" becomes the last day of December 2023)
- "YYYY-MM-DD" for a specific date
- NULL defaults to the current date (today)

language — Language for column names in the returned data.frame:

- "eng" (default): Returns columns date and selic_rate
- "pt": Returns columns data_referencia and taxa_selic

labels — Logical indicating whether to add variable labels using the labelled package. Labels provide descriptive text for each column when available.

Value

A data.frame with SELIC rate. Columns depend on the language parameter:

- English (language = "eng"): date (Date), selic_rate (numeric, % per year)
- Portuguese (language = "pt"): data_referencia (Date), taxa_selic (numeric, % ao ano)

Note

IMPORTANT API LIMITATION: The BCB API imposes a **10-year maximum window** for daily frequency series like SELIC. Requests spanning more than 10 years will fail. For longer historical analyses, split your request into multiple 10-year periods.

DEFAULT PERIOD: When start_date = NULL, defaults to "2020-01-01" (start of 2020), providing recent data while avoiding the 10-year API limit with current dates.

Examples

```
# Default: from 2020 to current date
df <- get_selic_rate()

# Specific period within 10-year limit
df2 <- get_selic_rate("2020-01-01", "2023-12-31")

# Last 5 years (respecting 10-year limit)
df3 <- get_selic_rate(start_date = "2019")

# Portuguese column names and labels
df4 <- get_selic_rate(language = "pt")

# Complete year analysis
df5 <- get_selic_rate("2018", "2023")
```

get_series_info *Get Series Information*

Description

Returns detailed metadata about a specific economic series.

Usage

```
get_series_info(series_id, field = NULL, language = c("pt", "en"))
```

Arguments

series_id	The series ID (SGS code)
field	Specific field to return (optional)
language	Language for returned labels. Either "pt" or "en".

Value

A list with series metadata or a single value

Examples

```
get_series_info("433")
get_series_info("433", field = "description", language = "en")
```

get_unemployment *Get Brazilian Unemployment Rate (PNAD Continua)*

Description

Downloads monthly unemployment rate data from the Brazilian Central Bank's SGS (Sistema Gerenciador de Series Temporais). The series corresponds to the unemployment rate from IBGE's Continuous PNAD survey (PNAD Continua), replicated and made available by the Central Bank.

Usage

```
get_unemployment(
  start_date = "2020-01-01",
  end_date = NULL,
  language = "eng",
  labels = TRUE
)
```

Arguments

start_date	Start date for the data period. Accepts multiple formats: • "YYYY" for year only (e.g., "2020" becomes "2020-01-01") • "YYYY-MM" for year and month (e.g., "2020-06" becomes "2020-06-01") • "YYYY-MM-DD" for a specific date • NULL defaults to "2020-01-01"
end_date	End date for the data period. Accepts the same formats as start_date. • NULL defaults to the current date
language	Language for column names in the returned data.frame: • "eng" (default): Returns columns date, unemployment_rate • "pt": Returns columns data, taxa_desemprego
labels	Logical indicating whether to add variable labels using the labelled package.

Value

A data.frame with:

- date (Date): Reference month
- value (numeric): Unemployment rate (%)

Note

Data Source: Brazilian Central Bank (SGS), series 24369. The data originates from IBGE's Continuous National Household Sample Survey (PNAD Continua) and is published with monthly frequency.

Although published monthly, the unemployment rate follows IBGE's moving-quarter methodology.

Examples

```
# Default: from 2020 to current date (aligned with other functions)
df <- get_unemployment()

# Specific period with year-only format
df2 <- get_unemployment("2018", "2023")

# Portuguese column names and labels
df3 <- get_unemployment(language = "pt")

# Without variable labels
df4 <- get_unemployment("2020-01-01", "2022-12-31", labels = FALSE)
```

plot_cdi_rate *Plot Brazilian CDI rate*

Description

Generates a time series plot of the CDI (Certificado de Depósito Interbancário) rate. The CDI is the benchmark interest rate for interbank deposits in Brazil and serves as a reference for many fixed income investments.

Usage

```
plot_cdi_rate(data, language = "eng")
```

Arguments

data	Tibble returned by get_cdi_rate(), with columns date and value.
language	Language for titles and labels: "pt" (Portuguese) or "eng" (English).

Value

A ggplot2 object showing the CDI rate over time.

Examples

```
# Example 1: English version
cdi_data <- get_cdi_rate(2020, 2024)
cdi_plot <- plot_cdi_rate(cdi_data)
print(cdi_plot)

# Example 2: Portuguese version
dados_cdi <- get_cdi_rate(2020, 2024, language = "pt")
grafico_cdi <- plot_cdi_rate(dados_cdi, language = "pt")
print(grafico_cdi)
```

plot_exchange_rate *Plot Brazilian exchange rate (USD/BRL)*

Description

Generates a time series plot of the USD/BRL exchange rate using data from get_exchange_rate(). Shows the commercial exchange rate for US Dollar to Brazilian Real.

Usage

```
plot_exchange_rate(data, language = "eng")
```

Arguments

data	Tibble returned by get_exchange_rate()
language	Language for titles and labels: "pt" (Portuguese) or "eng" (English).

Value

A ggplot2 object showing the exchange rate over time.

Examples

```
# Example 1: English version
exchange_data <- get_exchange_rate("2023-01-01", "2023-12-31")
exchange_plot <- plot_exchange_rate(exchange_data)
print(exchange_plot)

# Example 2: Portuguese version
dados_cambio <- get_exchange_rate("2023-01-01", "2023-12-31", language = "pt")
grafico_cambio <- plot_exchange_rate(dados_cambio, language = "pt")
print(grafico_cambio)
```

---

plot_inflation_rate *Plot Brazilian Inflation Rate (IPCA)*

Description

Generates a time series plot of Brazil's monthly inflation rate measured by the IPCA (Índice Nacional de Preços ao Consumidor Amplo). The IPCA is Brazil's official inflation index and is widely used for monetary policy decisions, contract indexation, and economic analysis.

Usage

```
plot_inflation_rate(data, language = "eng")
```

Arguments

data	Tibble returned by get_inflation_rate(), with columns date and value. The value column represents the monthly IPCA inflation rate (%).
language	Language for titles and labels: "pt" (Portuguese) or "eng" (English).

Value

A ggplot2 object showing the monthly inflation rate over time.

Examples

```
# Example 1: English version
inflation_data <- get_inflation_rate(2020, 2024)
inflation_plot <- plot_inflation_rate(inflation_data)
print(inflation_plot)

# Example 2: Portuguese version
dados_inflacao <- get_inflation_rate(2020, 2024, language = "pt")
grafico_inflacao <- plot_inflation_rate(dados_inflacao, language = "pt")
print(grafico_inflacao)
```

---

plot_selic_rate *Plot Brazilian SELIC rate (annualized, base 252)*

Description

Generates a time series plot of the SELIC interest rate using data from `get_selic()`. The SELIC rate ("Sistema Especial de Liquidação e de Custódia") represents the effective annualized rate (252-business-day basis) for overnight interbank loans and is the main instrument of Brazil's monetary policy.

Usage

```
plot_selic_rate(data, language = "eng")
```

Arguments

`data`	Tibble returned by `get_selic_rate()`
`language`	Language for titles and labels: "pt" (Portuguese) or "eng" (English).

Value

A `ggplot2` object showing the SELIC rate over time.

Examples

```
# Example 1: English version
selic_data <- get_selic_rate(2020, 2024)
selic_plot <- plot_selic_rate(selic_data)
print(selic_plot)

# Example 2: Portuguese version
dados_selic <- get_selic_rate(2020, 2024, language = "pt")
grafico_selic <- plot_selic_rate(dados_selic, language = "pt")
print(grafico_selic)
```

`plot_series_comparison` *Compare multiple financial/time series indices*

Description

Plots multiple time series on the same chart for comparison.

Usage

```
plot_series_comparison(
  data_list,
  y_vars,
  date_vars,
  language = "eng",
  scale_type = c("none", "index", "percent_change"),
  title = NULL,
  subtitle = NULL,
  y_label = NULL,
  caption = NULL,
  colors = NULL,
  line_types = NULL,
  show_legend = TRUE,
  legend_position = "bottom"
)
```

Arguments

`data_list`	Named list of data frames, each returned by a get_* function
`y_vars`	Vector of column names containing the values to plot from each data frame
`date_vars`	Vector of column names containing dates from each data frame
`language`	Language for labels: "pt" (Portuguese) or "eng" (English)
`scale_type`	Scaling applied to the series: **"none"** Plots raw values as provided **"index"** Indexes all series to 100 at the first observation **"percent_change"** Plots percentage change relative to the first observation
`title`	Plot title
`subtitle`	Plot subtitle
`y_label`	Y-axis label
`caption`	Plot caption
`colors`	Vector of colors for each series
`line_types`	Vector of line types for each series
`show_legend`	Whether to show the legend (default: TRUE)
`legend_position`	Position of legend ("bottom", "top", "left", "right", or "none")

Value

A ggplot2 object

Examples

```
# Example comparing multiple series
selic <- get_selic_rate(2020, 2024)
ipca <- get_ipca(2020, 2024)
igpm <- get_igpm(2020, 2024)

comparison_plot <- plot_series_comparison(
  data_list = list(SELIC = selic, IPCA = ipca, IGPM = igpm),
  y_vars = c("value", "value", "value"),
  date_vars = c("date", "date", "date"),
  scale_type = "index",
  title = "Comparison of Brazilian Economic Indicators",
  y_label = "Index (2020-01 = 100)",
  language = "eng"
)
print(comparison_plot)
```

---

plot_unemployment *Plot Brazil's monthly unemployment rate*

Description

Generates a ggplot2 line chart of Brazil's unemployment rate (PNAD Contínua) using data returned by `get_unemployment()`.

Usage

```
plot_unemployment(data, language = "eng")
```

Arguments

`data`	Tibble or data.frame returned by `get_unemployment()`. Must contain columns `date` (Date) and `value` (numeric).
`language`	Language of plot labels: • "eng" (default) • "pt"

Value

A ggplot2 object.

Examples

```
# Example 1: English version
unemployment_data <- get_unemployment("2020", "2024")
unemployment_plot <- plot_unemployment(unemployment_data)
print(unemployment_plot)

# Example 2: Portuguese version
dados_desemprego <- get_unemployment("2020", "2024")
grafico_desemprego <- plot_unemployment(dados_desemprego, language = "pt")
print(grafico_desemprego)
```

---

rule_of_114 *Rule of 114*

Description

Estimates the number of years required to triple an investment.

Usage

```
rule_of_114(rate)
```

Arguments

rate	Annual interest rate (decimal).

Value

Approximate years to triple.

Examples

```
rule_of_114(rate = 0.08)
```

---

rule_of_72 *Rule of 72*

Description

Estimates the number of years required to double an investment.

Usage

```
rule_of_72(rate)
```

Arguments

`rate`	Annual interest rate (decimal).

Value

Approximate years to double.

Examples

```
rule_of_72(rate = 0.08)
```

Index

∗ **datasets**
br_available_series, 3

br_available_series, 3
browse_series, 2

calc_amortization_schedule, 5
calc_compound_interest, 5
calc_continuous_compounding, 6
calc_effective_rate, 7
calc_future_value, 7
calc_future_value_ext, 8
calc_fv_annuity, 9
calc_irr, 9
calc_nominal_rate, 10
calc_nper, 10
calc_npv, 11
calc_pmt, 12
calc_present_value, 12
calc_pv_annuity, 13
calc_pv_continuous, 14
calc_rate, 14
calc_simple_interest, 15

get_cdi_rate, 16
get_exchange_rate, 17
get_gdp_growth, 19
get_inflation_rate, 20
get_selic_rate, 22
get_series_info, 24
get_unemployment, 24

plot_cdi_rate, 26
plot_exchange_rate, 26
plot_inflation_rate, 27
plot_selic_rate, 28
plot_series_comparison, 29
plot_unemployment, 30

rule_of_114, 31
rule_of_72, 31