

Package: deflateBR (via r-universe)

September 30, 2026

Type Package
Title Deflate Nominal Brazilian Reais
Version 1.2.0
Description Simple functions to deflate nominal Brazilian Reais using several popular price indexes downloaded from the Brazilian Institute for Applied Economic Research.
License MIT + file LICENSE
Depends R (>= 3.1.2)
Imports dplyr (>= 0.5.0), lubridate, http
Encoding UTF-8
LazyData true
URL <https://github.com/meirelesff/deflatebr/>
BugReports <https://github.com/meirelesff/deflatebr/issues/>
RoxygenNote 7.3.2
Suggests knitr, rmarkdown
Config/pak/sysreqs libssl-dev
Repository <https://meirelesff.r-universe.dev>
Date/Publication 2025-08-06 23:03:57 UTC
RemoteUrl <https://github.com/meirelesff/deflatebr>
RemoteRef HEAD
RemoteSha 5f02b85d830353a3f901f51c5fd5d8236581a8a2

Contents

.IPEA_BASE_URL	2
cache_exists	2
deflate	3
get_available_indexes	4
get_cache_dir	5

get_cache_file_path	5
get_price_index_data	6
get_price_index_data_cached	6
load_from_cache	7
save_to_cache	7

Index	9
--------------	----------

. IPEA_BASE_URL	<i>API Configuration and Data Fetching Functions</i>
-----------------	--

Description

Functions to handle API interactions with IPEA’s database for price index data.

Usage

. IPEA_BASE_URL

Format

An object of class character of length 1.

cache_exists	<i>Check if Cache File Exists and is Valid</i>
--------------	--

Description

Checks if a cache file exists for the given index and validates its structure.

Usage

cache_exists(index, verbose = TRUE)

Arguments

- | | |
|---------|---|
| index | Character string indicating the price index. |
| verbose | Logical indicating whether to show cache status messages. |

Value

Logical indicating whether a valid cache file exists.

deflate

*Deflate Nominal Brazilian Reais Using Various Price Indexes***Description**

deflate() is the main function of the deflateBR package. It uses data from the Brazilian Institute for Applied Economic Research's API (IPEADATA) to adjust nominal Brazilian Reais for inflation using various Brazilian price indexes.

Usage

```
deflate(
  nominal_values,
  nominal_dates,
  real_date,
  index = c("ipca", "igpm", "igpdi", "ipc", "inpc"),
  verbose = TRUE,
  cache = TRUE
)
```

Arguments

nominal_values	A numeric vector containing nominal Brazilian Reais to deflate.
nominal_dates	A Date vector with corresponding nominal dates (i.e., when nominal values were measured). Values are set to the previous month, following the standard methodology used by the Brazilian Central Bank .
real_date	A reference date to deflate nominal values. Can be either: • A character string in 'MM/YYYY' format (e.g., '01/2018' for January 2018) • A Date object with length 1 (e.g., as.Date("2018-01-01"))
index	Indicates the price index used to deflate nominal Reais. Valid options are: ipca, igpm, igpdi, ipc, and inpc.
verbose	Logical indicating whether to show progress messages and progress bar during data download. Default is TRUE.
cache	Logical indicating whether to use local caching to store downloaded price index data. When TRUE, data is saved locally and reused in subsequent calls, improving performance. Default is TRUE.

Details

Each one of the five price indexes included in the function are maintained by two Brazilian agencies: IPCA and INPC indexes are maintained by Brazilian Institute of Geography and Statistics (IBGE); IGP-M, IGP-DI, and IPC are maintained by Getulio Vargas Foundation (FGV). For an overview of the indexes' methodologies and covered periods, check the Brazilian Central Bank official [FAQ](#).

Value

A numeric vector.

References

For more information on the Brazilian Institute for Applied Economic Research's API, please check (in Portuguese): <http://www.ipeadata.gov.br/>.

Examples

```
## Not run:
# Use IPCA index to deflate a vector of nominal Brazilian Reais
reais <- rep(100, 5)
actual_dates <- seq.Date(from = as.Date("2001-01-01"), to = as.Date("2001-05-01"), by = "month")

# Using character string for reference date
deflate(reais, actual_dates, "01/2018", "ipca")

# Using Date object for reference date
deflate(reais, actual_dates, as.Date("2018-01-01"), "ipca")

# Using IGP-M index
deflate(reais, actual_dates, "01/2018", "igpm")

# Silent operation without progress messages
deflate(reais, actual_dates, as.Date("2018-01-01"), "ipca", verbose = FALSE)

# Disable caching (always download fresh data)
deflate(reais, actual_dates, "01/2018", "ipca", cache = FALSE)

## End(Not run)
```

get_available_indexes *Get Available Price Index Codes*

Description

Returns a named vector of available price index codes and their corresponding series codes.

Usage

```
get_available_indexes()
```

Value

A named character vector where names are index identifiers and values are series codes.

get_cache_dir	<i>Data Caching Functions</i>
---------------	-------------------------------

Description

Functions to handle local caching of price index data to improve performance by avoiding redundant API calls. Get Cache Directory Path

Returns the path to the cache directory, creating it if it doesn't exist.

Usage

```
get_cache_dir(create_if_missing = TRUE)
```

Arguments

create_if_missing	Logical indicating whether to create the directory if it doesn't exist.
-------------------	---

Value

Character string with the path to the cache directory.

get_cache_file_path	<i>Get Cache File Path for Index</i>
---------------------	--------------------------------------

Description

Returns the full path to the cache file for a specific price index.

Usage

```
get_cache_file_path(index)
```

Arguments

index	Character string indicating the price index.
-------	--

Value

Character string with the full path to the cache file.

get_price_index_data *Get Price Index Data from IPEA API*

Description

Fetches price index data from IPEA's API with retry mechanism and proper error handling.

Usage

```
get_price_index_data(index, max_retries = 3, retry_delay = 1, verbose = TRUE)
```

Arguments

index	Character string indicating the price index. Valid options are: "ipca", "igpm", "igpdi", "ipc", "inpc".
max_retries	Maximum number of retry attempts for failed requests. Default is 3.
retry_delay	Delay in seconds between retry attempts. Default is 1.
verbose	Logical indicating whether to show progress messages and progress bar. Default is TRUE.

Value

A data frame with columns VALDATA (dates) and VALVALOR (index values).

get_price_index_data_cached
Get Price Index Data with Caching Support

Description

Main function that handles cache logic - checks cache first, then downloads if needed.

Usage

```
get_price_index_data_cached(
  index,
  cache = TRUE,
  verbose = TRUE,
  max_retries = 3,
  retry_delay = 1
)
```

Arguments

index	Character string indicating the price index.
cache	Logical indicating whether to use caching.
verbose	Logical indicating whether to show progress messages.
max_retries	Maximum number of retry attempts for failed requests.
retry_delay	Delay in seconds between retry attempts.

Value

A data frame with columns VALDATA (dates) and VALVALOR (index values).

load_from_cache	<i>Load Data from Cache</i>
-----------------	-----------------------------

Description

Loads price index data from the local cache file.

Usage

```
load_from_cache(index, verbose = TRUE)
```

Arguments

index	Character string indicating the price index.
verbose	Logical indicating whether to show cache loading messages.

Value

A data frame with columns VALDATA (dates) and VALVALOR (index values).

save_to_cache	<i>Save Data to Cache</i>
---------------	---------------------------

Description

Saves price index data to the local cache file.

Usage

```
save_to_cache(index, data, verbose = TRUE)
```

Arguments

<code>index</code>	Character string indicating the price index.
<code>data</code>	Data frame with price index data to cache.
<code>verbose</code>	Logical indicating whether to show cache saving messages.

Value

Logical indicating success of the caching operation.

Index

* **datasets**

. IPEA_BASE_URL, [2](#)

. IPEA_BASE_URL, [2](#)

cache_exists, [2](#)

deflate, [3](#)

get_available_indexes, [4](#)

get_cache_dir, [5](#)

get_cache_file_path, [5](#)

get_price_index_data, [6](#)

get_price_index_data_cached, [6](#)

load_from_cache, [7](#)

save_to_cache, [7](#)