

Package ‘geocodebr’

October 8, 2025

Type Package

Title Geolocalização De Endereços Brasileiros (Geocoding Brazilian Addresses)

Version 0.3.0

Description Método simples e eficiente de geolocalizar dados no Brasil. O pacote é baseado em conjuntos de dados espaciais abertos de endereços brasileiros, utilizando como fonte principal o Cadastro Nacional de Endereços para Fins Estatísticos (CNEFE). O CNEFE é publicado pelo Instituto Brasileiro de Geografia e Estatística (IBGE), órgão oficial de estatísticas e geografia do Brasil. (A simple and efficient method for geolocating data in Brazil. The package is based on open spatial datasets of Brazilian addresses, primarily using the Cadastro Nacional de Endereços para Fins Estatísticos (CNEFE), published by the Instituto Brasileiro de Geografia e Estatística (IBGE), Brazil's official statistics and geography agency.)

License MIT + file LICENSE

URL <https://github.com/ipeaGIT/geocodebr>,
<https://ipeagit.github.io/geocodebr/>

BugReports <https://github.com/ipeaGIT/geocodebr/issues>

Depends R (>= 4.1.0)

Imports arrow (>= 15.0.1), checkmate, cli, data.table, DBI, dplyr, duckdb, enderecobr (>= 0.4.1), fs, glue, h3r, httr2 (>= 1.0.0), nanoarrow (>= 0.3.0.1), parallel, purrr, Rcpp, rlang, sf, sfheaders, tools

Suggests covr, dbplyr, geobr, ggplot2 (>= 3.3.1), knitr, rmarkdown, scales, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

LinkingTo Rcpp

NeedsCompilation yes

Author Rafael H. M. Pereira [aut, cre] (ORCID:
<https://orcid.org/0000-0003-2125-7465>),
 Daniel Herszenhut [aut] (ORCID:
<https://orcid.org/0000-0001-8066-1105>),
 Arthur Bazolli [ctb],
 Ipea - Instituto de Pesquisa Econômica Aplicada [cph, fnd]

Maintainer Rafael H. M. Pereira <rafa.pereira.br@gmail.com>

Repository CRAN

Date/Publication 2025-10-08 18:00:02 UTC

Contents

busca_por_cep	2
definir_campos	3
definir_pasta_cache	5
deletar_pasta_cache	6
download_cnefe	6
geocode	7
geocode_reverso	11
listar_dados_cache	13
listar_pasta_cache	14
Index	15

busca_por_cep	<i>Busca por CEP</i>
---------------	----------------------

Description

Busca endereços e suas coordenadas geográficas a partir de um CEP. As coordenadas de output utilizam o sistema de coordenadas geográficas SIRGAS 2000, EPSG 4674.

Usage

```
busca_por_cep(  
  cep,  
  h3_res = NULL,  
  resultado_sf = FALSE,  
  verboso = TRUE,  
  cache = TRUE  
)
```

Arguments

cep	Vetor. Um CEP ou um vetor de CEPs com 8 dígitos.
h3_res	Um número que indica a resolução espacial da célula hexagonal H3 da localização dos pontos retornados. Por padrão, é NULL. Detalhes sobre as resoluções disponíveis em https://h3geo.org/docs/core-library/restable/
resultado_sf	Lógico. Indica se o resultado deve ser um objeto espacial da classe sf. Por padrão, é FALSE, e o resultado é um data.frame com as colunas lat e lon.
verboso	Um valor lógico. Indica se barras de progresso e mensagens devem ser exibidas durante o download dos dados do CNEFE e a geocodificação dos endereços. O padrão é TRUE.
cache	Um valor lógico. Indica se os dados do CNEFE devem ser salvos ou lidos do cache, reduzindo o tempo de processamento em chamadas futuras. O padrão é TRUE. Quando FALSE, os dados do CNEFE são baixados para um diretório temporário.

Value

Retorna um data.frame com os CEPs de input e os endereços presentes naquele CEP com suas coordenadas geográficas de latitude (lat) e longitude (lon). Alternativamente, o resultado pode ser um objeto sf.

Examples

```
library(geocodebr)

# amostra de CEPs
ceps <- c("70390-025", "20071-001", "99999-999")

df <- geocodebr::busca_por_cep(
  cep = ceps,
  h3_res = 10,
  verboso = FALSE
)

head(df)
```

definir_campos

Especifica as colunas que descrevem os campos dos endereços

Description

Cria um vetor de caracteres especificando as colunas que representam cada campo do endereço na tabela de endereços. Os campos estado e municipio são obrigatórios.

Usage

```
definir_campos(  
    estado,  
    municipio,  
    logradouro = NULL,  
    numero = NULL,  
    cep = NULL,  
    localidade = NULL  
)
```

Arguments

estado	Uma string. O nome da coluna que representa o estado do endereço. Campo obrigatório. Na tabela de endereços, essa coluna pode conter os nomes dos estados por extenso, ou a abreviação oficial dos estados com duas letras, e.g. "AM", "SP", "DF", "RJ".
municipio	Uma string. O nome da coluna que representa o município do endereço. Campo obrigatório. Na tabela de endereços, essa coluna pode conter o nome dos municípios, ou o seu código IBGE de 7 dígitos.
logradouro	Uma string. O nome da coluna que representa o <i>logradouro</i> (endereço da rua) do endereço. Pode ser NULL se o campo não estiver especificado na tabela de endereços. Na tabela de endereços, essa coluna deve incluir o <i>tipo</i> do logradouro, indicando se trata-se de uma "Rua" ou "Avenida" etc, por exemplo "Avenida Presidente Getúlio Vargas". Além disso, essa coluna <i>não</i> deve incluir o número do endereço, pois o número deve ser indicado numa coluna separada.
numero	Uma string. O nome da coluna que representa o número do endereço. Pode ser NULL se o campo não estiver especificado na tabela de endereços. Na tabela de endereços, valores como 0 ou caracteres não numéricos como "S/N" ou "10a" são considerados como NA.
cep	Uma string. O nome da coluna que representa o <i>CEP</i> (Código de Endereçamento Postal) do endereço. Pode ser NULL se o campo não estiver especificado na tabela de endereços.
localidade	Uma string. O nome da coluna que representa a localidade (equivalente ao 'bairro' em áreas urbanas) do endereço. Pode ser NULL se esse campo não estiver presente na tabela de endereços.

Value

Um vetor de caracteres no qual os nomes são os campos do endereço e os valores são as colunas que os representam na tabela de endereços.

Examples

```
definir_campos(  
    logradouro = "Nome_logradouro",  
    numero = "Numero",  
    cep = "CEP",
```

```
localidade = "Bairro",  
municipio = "Cidade",  
estado = "UF"  
)
```

definir_pasta_cache *Define um diretório de cache para o geocodebr*

Description

Define um diretório de cache para os dados do geocodebr. Essa configuração é persistente entre sessões do R.

Usage

```
definir_pasta_cache(path, verboso = TRUE)
```

Arguments

path	Uma string. O caminho para o diretório usado para armazenar os dados em cache. Se NULL, o pacote usará um diretório versionado salvo dentro do diretório retornado por <code>tools::R_user_dir()</code> .
verboso	Um valor lógico. Indica se barras de progresso e mensagens devem ser exibidas durante o download dos dados do CNEFE e a geocodificação dos endereços. O padrão é TRUE.

Value

Retorna de forma invisível o caminho do diretório de cache.

Examples

```
definir_pasta_cache(tempdir())  
  
# retoma pasta padrão do pacote  
definir_pasta_cache( path = NULL)
```

deletar_pasta_cache *Deletar pasta de cache do geocodebr*

Description

Deleta todos arquivos da pasta do cache.

Usage

```
deletar_pasta_cache()
```

Value

Retorna de forma invisível o caminho do diretório de cache.

Examples

```
deletar_pasta_cache()
```

download_cnefe *Faz download dos dados do CNEFE*

Description

Faz o download de uma versão pre-processada e enriquecida do CNEFE (Cadastro Nacional de Endereços para Fins Estatísticos) que foi criada para o uso deste pacote.

Usage

```
download_cnefe(tabela = "todas", verboso = TRUE, cache = TRUE)
```

Arguments

tabela	Nome da tabela para ser baixada. Por padrão, baixa "todas".
verboso	Um valor lógico. Indica se barras de progresso e mensagens devem ser exibidas durante o download dos dados do CNEFE e a geocodificação dos endereços. O padrão é TRUE.
cache	Um valor lógico. Indica se os dados do CNEFE devem ser salvos ou lidos do cache, reduzindo o tempo de processamento em chamadas futuras. O padrão é TRUE. Quando FALSE, os dados do CNEFE são baixados para um diretório temporário.

Value

Retorna o caminho para o diretório onde os dados foram salvos.

Examples

```
download_cnefe(verboso = FALSE)
```

geocode

Geocaliza endereços no Brasil

Description

Geocodifica endereços brasileiros com base nos dados do CNEFE. Os endereços de input devem ser passados como um `data.frame`, no qual cada coluna descreve um campo do endereço (logradouro, número, cep, etc). Os resultados dos endereços geocalizados podem seguir diferentes níveis de precisão. Consulte abaixo a seção "Detalhes" para mais informações. As coordenadas de output utilizam o sistema de coordenadas geográficas SIRGAS 2000, EPSG 4674.

Usage

```
geocode(  
  enderecos,  
  campos_endereco = definir_campos(),  
  resultado_completo = FALSE,  
  resolver_empates = FALSE,  
  h3_res = NULL,  
  resultado_sf = FALSE,  
  verboso = TRUE,  
  cache = TRUE,  
  n_cores = 1  
)
```

Arguments

- `enderecos` Um `data.frame`. Os endereços a serem geocalizados. Cada coluna deve representar um campo do endereço.
- `campos_endereco` Um vetor de caracteres. A correspondência entre cada campo de endereço e o nome da coluna que o descreve na tabela `enderecos`. A função `definir_campos()` auxilia na criação deste vetor e realiza algumas verificações nos dados de entrada. Campos de endereço passados como `NULL` serão ignorados, e a função deve receber pelo menos um campo não nulo, além dos campos "estado" e "município", que são obrigatórios. Note que o campo "localidade" é equivalente a "bairro".
- `resultado_completo` Lógico. Indica se o output deve incluir colunas adicionais, como o endereço encontrado de referência. Por padrão, é `FALSE`.

<code>resolver_empates</code>	Lógico. Alguns resultados da geolocalização podem indicar diferentes coordenadas possíveis (e.g. duas ruas diferentes com o mesmo nome em uma mesma cidade). Esses casos são trados como 'empate' e o parâmetro <code>resolver_empates</code> indica se a função deve resolver esses empates automaticamente. Por padrão, é FALSE, e a função retorna apenas o caso mais provável. Para mais detalhes sobre como é feito o processo de desempate, consulte abaixo a seção "Detalhes".
<code>h3_res</code>	Um número que indica a resolução espacial da célula hexagonal H3 da localização dos pontos retornados. Por padrão, é NULL. Detalhes sobre as resoluções disponíveis em https://h3geo.org/docs/core-library/restable/
<code>resultado_sf</code>	Lógico. Indica se o resultado deve ser um objeto espacial da classe sf. Por padrão, é FALSE, e o resultado é um <code>data.frame</code> com as colunas <code>lat</code> e <code>lon</code> .
<code>verboso</code>	Um valor lógico. Indica se barras de progresso e mensagens devem ser exibidas durante o download dos dados do CNEFE e a geocodificação dos endereços. O padrão é TRUE.
<code>cache</code>	Um valor lógico. Indica se os dados do CNEFE devem ser salvos ou lidos do cache, reduzindo o tempo de processamento em chamadas futuras. O padrão é TRUE. Quando FALSE, os dados do CNEFE são baixados para um diretório temporário.
<code>n_cores</code>	Um número. O número de núcleos de CPU a serem utilizados no processamento dos dados. O padrão é 1.

Details

Precisão dos resultados:

A precisão dos resultados do **geocodebr** são apresentadas em 3 colunas, `precisao`, `tipo_resultado` e `desvio_metros`, explicadas abaixo.

Lidando com casos de empate:

No processo de geolocalização de dados, é possível que para alguns endereços de input sejam encontrados diferentes coordenadas possíveis (e.g. duas ruas diferentes com o mesmo nome, mas em bairros distintos em uma mesma cidade). Esses casos são trados como 'empate'. Quando a função `geocode()` recebe o o parâmetro `resolver_empates = TRUE`, os casos de empate são resolvidos automaticamente pela função. A solução destes empates é feita da seguinte maneira:

1. Quando se encontra diferente coordenadas possíveis para um endereço de input, nós assumimos que essas coordenadas pertencem provavelmente a endereços diferentes se (a) estas coordenadas estão a mais de 1Km entre si, ou (b) estão associadas a um logradouro 'ambíguo', i.e. que costumam se repetir em muitos bairros (e.g. "RUA A", "RUA QUATRO", "RUA 10", etc). Nestes casos, a solução de desempate é retornar o ponto com maior número de estabelecimentos no CNEFE, valor indicado na coluna "`contagem_cnefe`".
2. Quando as coordenadas possivelmente associadas a um endereço estão a menos de 1Km entre si e não se trata de um logradouro 'ambíguo', nós assumimos que os pontos pertencem provavelmente ao mesmo logradouro (e.g. diferentes CEPs ao longo de uma mesma rua). Nestes casos, a solução de desempate é retornar um ponto que resulta da média das coordenadas dos pontos possíveis ponderada pelo valor de "`contagem_cnefe`". Nesse caso, a coluna de output "`endereco_encontrado`" recebe valor do ponto com maior "`contagem_cnefe`".

Value

Retorna o `data.frame` de input endereços adicionado das colunas de latitude (`lat`) e longitude (`lon`), bem como as colunas (`precisao` e `tipo_resultado`) que indicam o nível de precisão e o tipo de resultado. Alternativamente, o resultado pode ser um objeto `sf`.

Precisão

Os resultados são classificados em seis amplas categorias de precisão:

1. "numero"
2. "numero_aproximado"
3. "logradouro"
4. "cep"
5. "localidade"
6. "municipio"

Cada nível de precisão pode ser desagregado em tipos de resultado mais refinados.

Tipos de resultados

A coluna `tipo_resultado` fornece informações mais detalhadas sobre como exatamente cada endereço de entrada foi encontrado no CNEFE. Em cada categoria, o **geocodebr** calcula a média da latitude e longitude dos endereços incluídos no CNEFE que correspondem ao endereço de entrada, com base em combinações de diferentes campos. No caso mais rigoroso, por exemplo, a função encontra uma correspondência determinística para todos os campos de um dado endereço ("estado", "municipio", "logradouro", "numero", "cep", "localidade"). Pense, por exemplo, em um prédio com vários apartamentos que correspondem ao mesmo endereço de rua e número. Nesse caso, as coordenadas dos apartamentos podem diferir ligeiramente, e o **geocodebr** calcula a média dessas coordenadas. Em um caso menos rigoroso, no qual apenas os campos ("estado", "municipio", "logradouro", "localidade") são encontrados, o **geocodebr** calcula as coordenadas médias de todos os endereços no CNEFE ao longo daquela rua e que se encontram na mesma localidade/bairro. Assim, as coordenadas de resultado tendem a ser o ponto médio do trecho daquela rua que passa dentro daquela localidade/bairro.

A coluna `tipo_resultado` fornece informações mais detalhadas sobre os campos de endereço utilizados no cálculo das coordenadas de cada endereço de entrada. Cada categoria é nomeada a partir de um código de quatro caracteres:

- o primeiro caracter, sempre `d` ou `p`, determina se a correspondência foi feita de forma determinística (`d`) ou probabilística (`p`);
- o segundo faz menção à categoria de precisão na qual o resultado foi classificado (`n` para "numero", `a` para "numero_aproximado", `r` para "logradouro", `c` para "cep", `b` para "localidade" e `m` para "municipio");
- o terceiro e o quarto caracteres designam a classificação de cada categoria dentro de seu grupo - via de regra, quanto menor o número formado por esses caracteres, mais precisas são as coordenadas calculadas.

As categorias de `tipo_resultado` são listadas abaixo, junto às categorias de `precisao` a qual elas estão associadas:

- precisao "numero"
 - dn01 - logradouro, numero, cep e localidade
 - dn02 - logradouro, numero e cep
 - dn03 - logradouro, numero e localidade
 - dn04 - logradouro e numero
 - pn01 - logradouro, numero, cep e localidade
 - pn02 - logradouro, numero e cep
 - pn03 - logradouro, numero e localidade
 - pn04 - logradouro e numero
- precisao "numero_aproximado"
 - da01 - logradouro, numero, cep e localidade
 - da02 - logradouro, numero e cep
 - da03 - logradouro, numero e localidade
 - da04 - logradouro e numero
 - pa01 - logradouro, numero, cep e localidade
 - pa02 - logradouro, numero e cep
 - pa03 - logradouro, numero e localidade
 - pa04 - logradouro e numero
- precisao "logradouro" (quando o número de entrada está faltando 'S/N')
 - dl01 - logradouro, cep e localidade
 - dl02 - logradouro e cep
 - dl03 - logradouro e localidade
 - dl04 - logradouro
 - pl01 - logradouro, cep e localidade
 - pl02 - logradouro e cep
 - pl03 - logradouro e localidade
 - pl04 - logradouro
- precisao "cep"
 - dc01 - municipio, cep, localidade
 - dc02 - municipio, cep
- precisao "localidade"
 - db01 - municipio, localidade
- precisao "municipio"
 - dm01 - municipio

Endereços não encontrados são retornados com latitude, longitude, precisão e tipo de resultado NA.

Desvio em metros

A coluna `desvio_metros` apresenta uma forma intuitiva e prática de saber o grau de incerteza do resultado encontrado. Essa coluna informa que pelo menos 95% de todos os pontos do CNEFE que possuem correspondência com o endereço de input estão num raio de x metros da localização encontrada.

Um desvio de até 30 metros, por exemplo, tende a representar um resultado muito confiável. A depender de como o dado geolocalizado será utilizado, até mesmos resultados com desvio_metros de até 500 ou 900 metros podem ser ser aceitáveis.

A coluna desvio_metros é particularmente útil para decidir por exemplo se um resultado encontrado com a precisão de CEP, localidade ou logradouro deveria ser aceitável. Por exemplo, muitas cidades do Brasil possuem um CEP único, o que tende a gerar resultados com alto grau de incerteza. Em várias cidades, no entanto, um CEP pode ser circunscrito a uma área muito pequena e as vezes até um único edifício. Nesses casos, o valor do desvio_metros tende a ser bem pequeno.

Busca probabilística

Os tipos de resultado com busca probabilística usam como base o algoritmo de semelhança de Jaro para comparar as strings de 'logradouro' dos dados de input e da base de endereços do geocodebr. O pacote considera como match o logradouro da base de endereços que apresenta a maior semelhança de Jaro condicionado a uma semelhança mínima, e desde que também haja match determinístico em ao menos um dos campos "cep" e "localidade". O geocodebr utiliza uma semelhança mínima de 0.85 nos casos de match probabilístico, e de 0.90 nos demais casos.

Examples

```
library(geocodebr)

# ler amostra de dados
data_path <- system.file("extdata/small_sample.csv", package = "geocodebr")
input_df <- read.csv(data_path)[1:2,]

fields <- geocodebr::definir_campos(
  logradouro = "nm_logradouro",
  numero = "Numero",
  cep = "Cep",
  localidade = "Bairro",
  municipio = "nm_municipio",
  estado = "nm_uf"
)

df <- geocodebr::geocode(
  enderecos = input_df,
  campos_endereco = fields,
  resolver_empates = TRUE,
  verboso = FALSE
)

head(df)
```

Description

Geocode reverso de coordenadas geográficas para endereços. A função recebe um `sf data frame` com pontos e retorna o endereço mais próximo dando uma distância máxima de busca.

Usage

```
geocode_reverso(  
  pontos,  
  dist_max = 1000,  
  verboso = TRUE,  
  cache = TRUE,  
  n_cores = 1  
)
```

Arguments

<code>pontos</code>	Uma tabela de dados com classe espacial <code>sf data frame</code> no sistema de coordenadas geográficas SIRGAS 2000, EPSG 4674.
<code>dist_max</code>	Integer. Distância máxima aceitável (em metros) entre os pontos de input e o endereço. Por padrão, a distância é de 1000 metros.
<code>verboso</code>	Um valor lógico. Indica se barras de progresso e mensagens devem ser exibidas durante o download dos dados do CNEFE e a geocodificação dos endereços. O padrão é TRUE.
<code>cache</code>	Um valor lógico. Indica se os dados do CNEFE devem ser salvos ou lidos do cache, reduzindo o tempo de processamento em chamadas futuras. O padrão é TRUE. Quando FALSE, os dados do CNEFE são baixados para um diretório temporário.
<code>n_cores</code>	Um número. O número de núcleos de CPU a serem utilizados no processamento dos dados. O padrão é 1.

Value

Retorna o `sf data.frame` de input adicionado das colunas do endereço encontrado. O output inclui uma coluna "distancia_metros" que indica a distância entre o ponto de input e o endereço mais próximo encontrado.

Examples

```
library(geocodebr)  
library(sf)  
  
# ler amostra de dados  
pontos <- readRDS(  
  system.file("extdata/pontos.rds", package = "geocodebr")  
)  
  
ponto <- pontos[1,]
```

```
# geocode reverso
df_enderecos <- geocodebr::geocode_reverso(
  pontos = ponto,
  dist_max = 800,
  verboso = TRUE,
  n_cores = 1
)

head(df_enderecos)
```

listar_dados_cache *Listar dados em cache*

Description

Lista os dados salvos localmente na pasta de cache

Usage

```
listar_dados_cache(print_tree = FALSE)
```

Arguments

`print_tree` Um valor lógico. Indica se o conteúdo da pasta de cache deve ser exibido em um formato de árvore. O padrão é FALSE.

Value

O caminho para os arquivos em cache

Examples

```
listar_dados_cache()

listar_dados_cache(print_tree = TRUE)
```

listar_pasta_cache *Obtém a pasta de cache usado no geocodebr*

Description

Obtém o caminho da pasta utilizada para armazenar em cache os dados do geocodebr. Útil para inspecionar a pasta configurada com `definir_pasta_cache()` em uma sessão anterior do R. Retorna a pasta de cache padrão caso nenhuma pasta personalizado tenha sido configurada anteriormente.

Usage

```
listar_pasta_cache()
```

Value

O caminho da pasta de cache.

Examples

```
listar_pasta_cache()
```

Index

`busca_por_cep`, 2

`definir_campos`, 3

`definir_campos()`, 7

`definir_pasta_cache`, 5

`definir_pasta_cache()`, 14

`deletar_pasta_cache`, 6

`download_cnefe`, 6

`geocode`, 7

`geocode_reverso`, 11

`listar_dados_cache`, 13

`listar_pasta_cache`, 14

`tools::R_user_dir()`, 5