Package: fullmetaphone (via r-universe)

October 15, 2026

Type Package

Title Full-Length Metaphone and Double Metaphone Phonetic Codes

Version 0.1.0

Description Encodes names and words as phonetic codes, so that names which sound alike but are spelled differently can be matched. Implements the Metaphone algorithm of Philips (1990, Computer Language, 7(12), 38-43) and the Double Metaphone algorithm of Philips (2000) <https://web.archive.org/web/20250702064845/https://drdobbs.com/the-double-metaphone-search-algorithm/184401251>, which gives a primary and an alternate code for names of many language origins. Unlike most implementations, including 'PGRdup', the codes are not cut to four characters, which keeps long and multi-part names apart. Provides metaphone(), double_metaphone() and sounds_like() for record linkage, deduplication and name search, and includes a data set of common United States surnames.

License GPL-3

URL https://github.com/anilyayavar/dblmetaphone

BugReports https://github.com/anilyayavar/dblmetaphone/issues

Copyright See file inst/COPYRIGHTS.

Depends R (>= 3.5.0)

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

LazyData true

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Anil Kumar Goyal [aut, cre, cph], Atharv Tyagi [aut, cph], Lawrence Philips [ctb] (Designer of the Metaphone and Double Metaphone algorithms and author of their original code), Maurice Aubrey [ctb] (Author of the C implementation of Double Metaphone whose rules are followed here), Kevin Atkinson [ctb] (Bug fixes to that C implementation), The Apache Software Foundation [cph] (Apache Commons Codec, the basis of metaphone() and of the test data)

Maintainer Anil Kumar Goyal <anilyayavar@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-15 17:00:07 UTC

RemoteUrl https://github.com/cran/fullmetaphone

RemoteRef HEAD

RemoteSha 2abec2a72e49652347e9b140630ff6186a6640ff

Contents

double_metaphone 2
metaphone 4
sounds_like 6
us_surnames 7

Index **9**

`double_metaphone` *Full-length Double Metaphone codes*

Description

Encodes names or words with the Double Metaphone algorithm of Lawrence Philips (2000). Each input gets two codes. The primary code is the most likely pronunciation. The secondary (alternate) code allows for another common pronunciation, for example of a name from another language.

Usage

```
double_metaphone(x, max_length = 32, by_word = FALSE)
```

Arguments

`x`	A character vector of names or words. Factors are accepted.
`max_length`	The largest number of characters to keep in each code, a single number of at least 1. The default is 32. Use ‘Inf‘ for no limit, or ‘4‘ for the traditional short codes. With ‘by_word = TRUE‘ the limit applies to each word separately. See the section on code length.
`by_word`	If ‘TRUE‘, encode each word separately and join the codes with spaces. See the section on several words.

Details

Most implementations, including ‘PGRdup::DoubleMetaphone()‘, keep only the first four characters of each code. Here codes keep up to 32 characters by default, so "Christopher Anderson" and "Christina Andrews" no longer get the same code. See the section on code length.

Value

A data frame with one row for each element of ‘x‘ and two character columns, ‘primary‘ and ‘secondary‘. Missing values in ‘x‘ give ‘NA‘ in both columns. A value with no codable letters gives empty strings.

Code length

‘max_length‘ sets the largest number of characters kept in each code. The default, 32, keeps the complete code for practically every real name, because a code has roughly one character per consonant sound. Only very long strings, such as a full name with many parts, reach the limit. Use ‘max_length = Inf‘ for no limit at all. Use ‘max_length = 4‘ for the traditional short codes, for example to compare results with other software. Codes made with different lengths should not be compared with each other.

The rules follow the C implementation by Maurice Aubrey in the Perl module Text::DoubleMetaphone, which is based on Philips’ own C++ code and is the version used by most other software. Cut to four characters, the codes agree with the reference data of Apache Commons Codec.

Preparing the text

Letters are changed to capitals, and common accented Latin letters are replaced by plain ones, so a u with an umlaut is read as U. The C with a cedilla and the N with a tilde keep their own Double Metaphone rules and are read as S and N. Apostrophes are removed, so "O’Brien" is read as "OBRIEN". Any other character that is not a letter, such as a digit, hyphen or full stop, is treated as a space. Names written in other scripts, such as Arabic, Chinese, Cyrillic or Devanagari, should be transliterated to Latin letters first.

Several words

By default the whole text is encoded as one string, exactly as the original algorithm does. Spaces are kept and some rules use them, for example "San Jacinto" or "Van Damme". The code then runs the words together. Set ‘by_word = TRUE‘ to encode each word on its own instead. The word codes are then joined with single spaces, so "Maria Gonzalez" gives ‘"MR KNSLS"‘. This makes it easy to compare the parts of names separately, or to compare names whose parts are written in a different order.

References

Philips, L. (2000). The double metaphone search algorithm. *C/C++ Users Journal*, 18(6), 38-43. https://web.archive.org/web/20250702064845/https://drdobbs.com/the-double-metaphone-search-algorith
184401251

Aubrey, M. Text::DoubleMetaphone, a Perl module. https://metacpan.org/pod/Text::DoubleMetaphone

See Also

[metaphone()] for the original single-code algorithm, [sounds_like()] to compare two sets of names, and ‘vignette("fullmetaphone")‘ for worked examples.

Examples

```
double_metaphone(c("Smith", "Schmidt", "Meyer", "Maier", "Xavier"))

# Full codes keep long names apart
people <- c("Christopher Anderson", "Christina Andrews")
double_metaphone(people)
double_metaphone(people, max_length = 4)

# Encode each part of a name separately
double_metaphone("Maria Gonzalez Lopez", by_word = TRUE)

# Add the codes to a data frame
staff <- data.frame(name = c("Stephen Phillips", "Steven Philips"))
cbind(staff, double_metaphone(staff$name))
```

---

metaphone *Full-length Metaphone codes*

Description

Encodes names or words with the original Metaphone algorithm of Lawrence Philips (1990). Each input gets one code. Words that sound alike in English, such as "Knight" and "Night", get the same code.

Usage

```
metaphone(x, max_length = 32, by_word = FALSE)
```

Arguments

x	A character vector of names or words. Factors are accepted.
max_length	The largest number of characters to keep in each code, a single number of at least 1. The default is 32. Use ‘Inf‘ for no limit, or ‘4‘ for the traditional short codes. With ‘by_word = TRUE‘ the limit applies to each word separately. See the section on code length.
by_word	If ‘TRUE‘, encode each word separately and join the codes with spaces. See the section on several words.

Details

Many implementations cut the code to four characters. Here codes keep up to 32 characters by default. Set ‘max_length = Inf‘ for no limit, or ‘max_length = 4‘ for the traditional short codes.

The rules follow the Metaphone implementation in Apache Commons Codec, a widely used reference version. Unlike Double Metaphone, Metaphone keeps a leading vowel as itself, so "Anna" gives ‘"AN"‘ and "Emma" gives ‘"EM"‘.

Value

A character vector of codes, the same length as ‘x‘. Missing values in ‘x‘ give ‘NA‘. A value with no codable letters gives an empty string.

Preparing the text

Letters are changed to capitals, and common accented Latin letters are replaced by plain ones. The C with a cedilla becomes S. Apostrophes are removed, and any other character that is not a letter is treated as a space. Names written in other scripts should be transliterated to Latin letters first.

Code length

‘max_length‘ sets the largest number of characters kept in each code. The default, 32, keeps the complete code for practically every real name, because a code has roughly one character per consonant sound. Only very long strings, such as a full name with many parts, reach the limit. Use ‘max_length = Inf‘ for no limit at all. Use ‘max_length = 4‘ for the traditional short codes, for example to compare results with other software. Codes made with different lengths should not be compared with each other.

The rules follow the C implementation by Maurice Aubrey in the Perl module Text::DoubleMetaphone, which is based on Philips’ own C++ code and is the version used by most other software. Cut to four characters, the codes agree with the reference data of Apache Commons Codec.

Several words

By default the whole text is encoded as one string, exactly as the original algorithm does. Spaces are kept and some rules use them, for example "San Jacinto" or "Van Damme". The code then runs the words together. Set ‘by_word = TRUE‘ to encode each word on its own instead. The word codes are then joined with single spaces, so "Maria Gonzalez" gives ‘"MR KNSLS"‘. This makes it easy to compare the parts of names separately, or to compare names whose parts are written in a different order.

References

Philips, L. (1990). Hanging on the metaphone. *Computer Language*, 7(12), 38-43.

Apache Commons Codec, the Metaphone class. https://commons.apache.org/proper/commons-codec/

See Also

[double_metaphone()] for the improved two-code algorithm, [sounds_like()] to compare two sets of names, and ‘vignette("fullmetaphone")‘ for worked examples.

Examples

```
metaphone(c("Knight", "Night", "Philip", "Filip", "Catherine", "Katherine"))

# Full codes against the traditional four characters
metaphone(c("Rosenberg", "Rosenbaum"))
metaphone(c("Rosenberg", "Rosenbaum"), max_length = 4)

metaphone("Jean Pierre Dubois", by_word = TRUE)
```

---

sounds_like *Do two names sound alike?*

Description

Compares two vectors of names element by element and reports whether each pair gets the same phonetic code.

Usage

```
sounds_like(
  x,
  y,
  method = c("double", "metaphone"),
  max_length = 32,
  by_word = FALSE
)
```

Arguments

x, y	Character vectors of names. They are recycled to a common length in the usual way, so one of them may be a single name.
method	Either ‘"double"‘ (the default) for Double Metaphone or ‘"metaphone"‘ for the original Metaphone.
max_length	The largest number of characters to keep in each code, a single number of at least 1. The default is 32. Use ‘Inf‘ for no limit, or ‘4‘ for the traditional short codes. With ‘by_word = TRUE‘ the limit applies to each word separately. See the section on code length.
by_word	If ‘TRUE‘, encode each word separately and join the codes with spaces. See the section on several words.

Details

With ‘method = "double"‘, a pair matches when any Double Metaphone code of one name (primary or secondary) equals any code of the other. This is the comparison Philips recommended. With ‘method = "metaphone"‘, the single Metaphone codes must be equal.

A name with no codable letters, such as ‘""‘ or ‘"123"‘, never matches.

Value

A logical vector. ‘NA‘ where either name is missing.

See Also

[double_metaphone()], [metaphone()], and ‘vignette("fullmetaphone")‘ for worked examples of deduplication and record linkage.

Examples

```
sounds_like("Meyer", c("Meier", "Mayer", "Maier", "Miller"))

sounds_like("Smith", "Schmidt")
sounds_like("Smith", "Schmidt", method = "metaphone")

# Full codes tell long names apart. Four-character codes do not.
sounds_like("Christopher Anderson", "Christina Andrews")
sounds_like("Christopher Anderson", "Christina Andrews", max_length = 4)

# Search a register of names
hits <- us_surnames[sounds_like("Schneider", us_surnames$surname), ]
head(hits)
```

---

us_surnames *Common surnames in the United States*

Description

The 5000 most common surnames recorded in the 2010 United States Census, with their frequencies. The list mixes names of English, German, Irish, Italian, Polish, Spanish, Portuguese, Chinese, Vietnamese, Korean, Indian and many other origins, which makes it a convenient test bed for phonetic matching.

Usage

```
us_surnames
```

Format

A data frame with 5000 rows and 3 columns.

surname The surname, in title case.

rank Rank by frequency. Surnames with equal counts share a rank.

count Number of people in the 2010 Census with this surname.

Details

The Census Bureau publishes surnames in capital letters. Here they are converted to title case with [tools::toTitleCase()], so "MCDONALD" becomes "Mcdonald". The capitalisation has no effect on the codes.

Source

United States Census Bureau, "Frequently Occurring Surnames from the 2010 Census", <https://www.census.gov/topics/popu
Data published by the Census Bureau are in the public domain.

Examples

```
head(us_surnames)

# Surnames that sound like Meyer, with how common each one is
us_surnames[sounds_like("Meyer", us_surnames$surname), ]
```

Index

∗ **datasets**
    us_surnames, 7

double_metaphone, 2

metaphone, 4

sounds_like, 6

us_surnames, 7