Package ‘medicalcoder’

February 22, 2026

Title A Unified and Longitudinally Aware Framework for ICD-Based Comorbidity Assessment

Version 0.8.0

Description Provides comorbidity classification algorithms such as the Pediatric Complex Chronic Conditions (PCCC), Charlson, and Elixhauser indices, supports longitudinal comorbidity flagging across encounters, and includes utilities for working with medical coding schemas such as the International Classification of Diseases (ICD).

Depends R (>= 3.5.0)

License BSD_3_clause + file LICENSE

Language en-US

Encoding UTF-8

URL `http://www.peteredewitt.com/medicalcoder/`, `https://github.com/dewittpe/medicalcoder/`

BugReports `https://github.com/dewittpe/medicalcoder/issues`

LazyData true

Suggests data.table, dplyr, kableExtra, knitr, R.utils, rmarkdown

RoxygenNote 7.3.3

VignetteBuilder knitr

Config/Needs/website pkgdown, comorbidity, icdcomorbid, multimorbidity, dplyr, odbc, DBI, RSQLite, pccc

NeedsCompilation no

Author Peter DeWitt [aut, cre, cov] (ORCID: <`https://orcid.org/0000-0002-6391-0795`>), Tell Bennett [ctb] (ORCID: <`https://orcid.org/0000-0003-1483-4236`>), Seth Russell [ctb] (ORCID: <`https://orcid.org/0000-0002-2436-1367`>), Meg Rebull [ctb] (ORCID: <`https://orcid.org/0000-0003-0334-4223`>), Vincent Rubinetti [cov] (ORCID: <`https://orcid.org/0000-0002-4655-3773`>)

Maintainer Peter DeWitt <`peter.dewitt@cuanschutz.edu`>

Repository CRAN

Date/Publication 2026-02-22 10:00:02 UTC

Contents

`comorbidities` *Comorbidities*

Description

Apply established comorbidity algorithms to ICD-coded data. Supported methods include several variants of the Charlson comorbidity system, Elixhauser, and the Pediatric Complex Chronic Conditions (PCCC).

Usage

```
comorbidities(
  data,
  icd.codes,
  method,
  id.vars = NULL,
  icdv.var = NULL,
  icdv = NULL,
  dx.var = NULL,
  dx = NULL,
  poa.var = NULL,
  poa = NULL,
  age.var = NULL,
  primarydx.var = NULL,
  primarydx = NULL,
  flag.method = c("current", "cumulative"),
  full.codes = TRUE,
  compact.codes = TRUE,
  subconditions = FALSE
)
```

Arguments

`data`	A `data.frame` in a "long" format. The input `data.frame` is expected to have one column of ICD codes (one code per row) with additional (optional) columns for patient/encounter ids, ICD version, diagnostic/procedure status, present-on-admission flags, primary diagnostic flags, or age.
`icd.codes`	Character scalar naming the column in `data` that contains ICD codes (character strings). Codes may be provided in full form (with decimal points, e.g., C84.2), compact form (dots omitted, e.g., C842), or any mix of the two. Matching against lookup tables is governed by `icdv.var`/`icdv`, `dx.var`/`dx`, and the `full.codes` / `compact.codes` flags.
`method`	Character string indicating the comorbidity algorithm to apply to `data`.
`id.vars`	Optional character vector of column names. When missing, the entire input `data` is treated as a single encounter from a single patient. If you want to set `flag.method` = `"cumulative"` then `length(id.vars)` >= `2` is expected. The last element should be the encounter order (must be sortable).
`icdv.var`	Character scalar naming the column in `data` that indicates the ICD version (9 or 10). If present it must be integer values `9` or `10`. `icdv.var` takes precedence over `icdv` if both are provided.
`icdv`	An integer value of `9L` or `10L` indicating that all `data[[icd.codes]]` are ICD version 9 or 10, respectively. Ignored (with a warning) if `icdv.var` is provided.
`dx.var`	Character scalar naming the column in `data` that indicates diagnostic (`1`) vs procedural (`0`) codes. If present it must be integer values `0` or `1`. `dx.var` takes precedence over `dx` if both are provided.
`dx`	An integer indicating that all `data[[icd.codes]]` are diagnostic (`1`) or procedure (`0`) codes. Ignored (with a warning) if `dx.var` is provided.
`poa.var`	Character scalar naming the column with present-on-admission flags: integer `1L` (present), `0L` (not present), or `NA`. PCCC and Charlson will only flag conditions when the code is present-on-admission. Elixhauser has a mix of conditions; some require present-on-admission while others do not. `poa.var` takes precedence over `poa` if both are provided.
`poa`	Integer scalar `0` or `1`. Use when all `icd.codes` share the same present-on-admission status. Ignored with a warning if `poa` and `poa.var` are both provided.
`age.var`	Character scalar naming the column in `data` that contains patient age in years. Only applicable to Charlson comorbidities.
`primarydx.var`	Character scalar naming the column in `data` that indicates whether `data[[icd.codes]]` are primary diagnostic codes (`1L`) or not (`0L`). Primary diagnosis is used only for

Elixhauser and Charlson comorbidities and is ignored when the method is a PCCC variant. `primarydx.var` takes precedence over `primarydx` if both are provided.

`primarydx`	An integer value of `0` or `1`. If `0`, treat all codes as non-primary diagnoses; if `1`, treat all codes as primary diagnoses. Ignored, with a warning, if `primarydx.var` is provided.
`flag.method`	When `flag.method = 'current'` (default) only codes associated with the current `id.vars` are considered when flagging comorbidities. When `flag.method = 'cumulative'` then all prior encounters are considered when flagging comorbidities. See **Details**.
`full.codes, compact.codes`	Logical; when `TRUE` compare `data[[icd.codes]]` against full and/or compact ICD codes in the method's lookup tables. Full ICD codes include a decimal point (when applicable) and compact codes omit the decimal point. For example: `B95.0` is the full ICD-10-CM diagnostic code for "Streptococcus, group A, as the cause of disease classified elsewhere," whereas `B950` is the associated compact code.
`subconditions`	Logical scalar; when `TRUE`, report both conditions and subconditions (PCCC only).

Details

When `flag.method = "current"`, only codes from the index encounter contribute to flags. When a longitudinal method is selected (e.g., `"cumulative"`), prior encounters for the same `id.vars` combination may contribute to condition flags. For the cumulative method to work, `id.vars` needs to be a character vector of length 2 or more. The last element is treated as the encounter identifier and must be sortable. For example, say you have data with a hospital, patient, and encounter id. The `id.vars` could be one of two entries: `c("hospital", "patient", "encounter")` or `c("patient", "hospital", "encounter")`. In both cases the return will be the same because the encounter identifier is unchanged regardless of whether hospital or patient is listed first.

It is critically important that the `data[[tail(id.vars, 1)]]` variable can be sorted. Just because your data is sorted in temporal order does not mean that the results will be correct if the `tail(id.vars, 1)` is not in the same order as the data. For example, say you had the following:

patid	enc_id	date
P1	10823090	Aug 2023
P1	10725138	Jul 2025

`id.vars = c("patid", "enc_id")` will give the wrong result as enc_id 10725138 would be sorted to come before enc_id 10823090. `id.vars = c("patid", "date")` would be sufficient input, assuming that `date` has been correctly stored. Adding a column `enc_seq`, e.g.,

patid	enc_id	date	enc_seq
P1	10823090	Aug 2023	1
P1	10725138	Jul 2025	2

and calling `comorbidities()` with `id.vars = c("patid", "enc_seq")` will have better performance than using the date and will clear up any possible issues with non-sequential encounter ids from the source data.

Cumulative + POA defaults:

When `flag.method = "cumulative"` and neither `poa` nor `poa.var` is supplied, the first encounter for a condition is treated as `poa = 0`. Subsequent encounters for that condition are flagged as `poa = 1`.

When `flag.method = "current"` and neither `poa` nor `poa.var` is supplied, then all codes will be considered present-on-admission. If poa was assumed to be 0, then in this case the only conditions that could be flagged are the Elixhauser conditions which are poa-exempt.

Value

The return object will be slightly different depending on the value of `method` and `subconditions`.

- When `subconditions = FALSE`, a `medicalcoder_comorbidities` object (a `data.frame` with attributes) is returned. Column(s) for `id.vars`, if defined in the function call. For all methods there will be the following columns:
  - `num_cmrb` a count of comorbidities/conditions flagged
  - `cmrb_flag` a 0/1 integer indicator for at least one comorbidity/condition.

  Additional columns:
  - PCCC methods:
    * For `method = "pccc_v2.0"` and `method = "pccc_v2.1"`, there is one indicator column per condition.
    * For `method = "pccc_v3.0"` and `method = "pccc_v3.1"`, there are four columns per condition:
      · `<condition>_dxpr_or_tech`: the condition was flagged due to the presence of either a diagnostic or procedure code, or was flagged due to the presence of a technology dependence code along with at least one comorbidity being flagged by a diagnostic or procedure code.
      · `<condition>_dxpr_only`: the condition was flagged due to the presence of a non-technology dependent diagnostic or procedure code only.
      · `<condition>_tech_only`: the condition was flagged due to the presence of a technology dependent code only and at least one other comorbidity was flagged by a non-technology dependent code.
      · `<condition>_dxpr_and_tech`: The patient had both diagnostic or procedure codes and a technology dependence code for the condition.
  - For Charlson variants, indicator columns are returned for the relevant conditions, `cci` (Charlson Comorbidity Index), and `age_score`.
  - For Elixhauser variants, indicator columns are returned for all relevant comorbidities, mortality, and readmission indices.
- When `subconditions = TRUE` and the method is a PCCC variant, a list of length two is returned: the first element contains condition indicators; the second element is a named list of `data.frames` with indicators for subconditions within each condition.

References

- Pediatric Complex Chronic Conditions:
  - Feudtner, C., Feinstein, J.A., Zhong, W. et al. Pediatric complex chronic conditions classification system version 2: updated for ICD-10 and complex medical technology dependence and transplantation. BMC Pediatr 14, 199 (2014). https://doi.org/10.1186/1471-2431-14-199
  - Feinstein JA, Hall M, Davidson A, Feudtner C. Pediatric Complex Chronic Condition System Version 3. JAMA Netw Open. 2024;7(7):e2420579. https://doi.org/10.1001/jamanetworkopen.2024.20579
- Charlson Comorbidities:
  - Mary E. Charlson, Peter Pompei, Kathy L. Ales, C.Ronald MacKenzie, A new method of classifying prognostic comorbidity in longitudinal studies: Development and validation, Journal of Chronic Diseases, Volume 40, Issue 5, 1987, Pages 373-383, ISSN 0021-9681, https://doi.org/10.1016/0021-9681(87)90171-8.
  - Deyo RA, Cherkin DC, Ciol MA. Adapting a clinical comorbidity index for use with ICD-9-CM administrative databases. J Clin Epidemiol. 1992 Jun;45(6):613-9. https://doi.org/10.1016/0895-4356(92)90133-8. PMID: 1607900.
  - Quan H, Sundararajan V, Halfon P, Fong A, Burnand B, Luthi JC, Saunders LD, Beck CA, Feasby TE, Ghali WA. Coding algorithms for defining comorbidities in ICD-9-CM and ICD-10 administrative data. Med Care. 2005 Nov;43(11):1130-9. https://doi.org/10.1097/01.mlr.0000182534.1983 PMID: 16224307.
  - Quan H, Li B, Couris CM, Fushimi K, Graham P, Hider P, Januel JM, Sundararajan V. Updating and validating the Charlson comorbidity index and score for risk adjustment in hospital discharge abstracts using data from 6 countries. Am J Epidemiol. 2011 Mar 15;173(6):676-82. https://doi.org/10.1093/aje/kwq433. Epub 2011 Feb 17. PMID: 21330339.
  - Glasheen WP, Cordier T, Gumpina R, Haugh G, Davis J, Renda A. Charlson Comorbidity Index: ICD-9 Update and ICD-10 Translation. Am Health Drug Benefits. 2019 Jun-Jul;12(4):188-197. PMID: 31428236; PMCID: PMC6684052.
- Elixhauser Comorbidities:
  - Agency for Healthcare Research and Quality (AHRQ). Elixhauser Comorbidity Software Refined for ICD-10-CM Diagnoses, v2026.1 [Internet]. 2026. Available from: https://www.hcup-us.ahrq.gov/toolssoftware/comorbidityicd10/comorbidity_icd10.jsp

See Also

- `vignettes(topic = "comorbidities", package = "medicalcoder")`
- `vignettes(topic = "pccc", package = "medicalcoder")`
- `vignettes(topic = "charlson", package = "medicalcoder")`
- `vignettes(topic = "elixhauser", package = "medicalcoder")`

Examples

```
pccc_v3.1_results <-
  comorbidities(data = mdcr,
                icd.codes = "code",
                id.vars = "patid",
                dx.var = "dx",
                method = "pccc_v3.1",
                flag.method = 'current',
                poa = 1)
summary(pccc_v3.1_results)

pccc_v3.1_subcondition_results <-
  comorbidities(data = mdcr,
                icd.codes = "code",
                id.vars = "patid",
                dx.var = "dx",
                method = "pccc_v3.1",
                flag.method = 'current',
                poa = 1,
                subconditions = TRUE)
summary(pccc_v3.1_subcondition_results)

charlson_results <-
  comorbidities(data = mdcr,
                icd.codes = "code",
                id.vars = "patid",
                dx.var = "dx",
                method = "charlson_quan2011",
                flag.method = 'current',
                poa = 1)
summary(charlson_results)

elixhauser_results <-
  comorbidities(data = mdcr,
                icd.codes = "code",
                id.vars = "patid",
                dx.var = "dx",
                method = "elixhauser_ahrq2025",
                primarydx = 1,
                flag.method = 'current',
                poa = 1)
summary(elixhauser_results)
```

get_charlson_codes *Get Charlson Codes*

Description

Retrieve a copy of internal lookup tables for the ICD codes used in assessing Charlson comorbidities.

Usage

```
get_charlson_codes()
```

Value

A `data.frame` with the following columns:

- `icdv`: Integer vector indicating if the code is from ICD-9 or ICD-10
- `dx`: Integer vector. 1 if the code is a diagnostic, (ICD-9-CM, ICD-10-CM, WHO, CDC Mortality), or 0 if the code is procedural (ICD-9-PCS, ICD-10-PCS)
- `full_code`: Character vector with the ICD code and any relevant decimal point
- `code`: Character vector with the compact ICD code
- `condition`: Character vector of the conditions
- `charlson_\<variant\>`: Integer vector indicating if the code is part of the \<variant\> of the Charlson comorbidities.

See Also

- `get_charlson_index_scores()` for a lookup table of the by comorbidity index scores.
- `get_icd_codes()` for the lookup table of all ICD codes.
- `get_pccc_codes()` for the lookup table of ICD codes used for the PCCC.
- `get_elixhauser_codes()` for the lookup table of ICD codes used for the Elixhauser comorbidities.
- `comorbidities()` for applying comorbidity algorithms to a data set.

Examples

```
head(get_charlson_codes())
str(get_charlson_codes())
```

`get_charlson_index_scores`

Get Charlson Index Scores

Description

Retrieve a copy of internal lookup tables of index scores used in assessing Charlson comorbidities.

Usage

```
get_charlson_index_scores()
```

Value

A `data.frame` with the following columns:

- `condition`: Character vector of the conditions
- `index`: Character vector indicating if the score is for the mortality or the readmission index score
- `charlson_<variant>`: the index scores for the variant

See Also

- `get_charlson_codes()` for a lookup table of the ICD codes mapping to the Charlson comorbidities.
- `comorbidities()` for applying comorbidity algorithms to a data set.

Examples

```
head(get_charlson_index_scores())
str(get_charlson_index_scores())
```

`get_elixhauser_codes` *Get Elixhauser Codes*

Description

Retrieve copy of internal lookup tables for the ICD codes used in assessing Elixhauser comorbidities.

Usage

```
get_elixhauser_codes()
```

Value

A `data.frame` with the following columns:

- `icdv`: Integer vector indicating if the code is from ICD-9 or ICD-10
- `dx`: Integer vector. 1 if the code is a diagnostic, (ICD-9-CM, ICD-10-CM, WHO, CDC Mortality), or 0 if the code is procedural (ICD-9-PCS, ICD-10-PCS)
- `full_code`: Character vector with the ICD code and any relevant decimal point
- `code`: Character vector with the compact ICD code omitting any relevant decimal point
- `condition`: Character vector of the conditions
- `elixhauser_<variant>`: Integer vector indicating if the code is part of the `<variant>` of the Elixhauser comorbidities.

See Also

- `get_elixhauser_index_scores()` for the lookup table of the condition by condition scores for mortality and readmission indices.
- `get_elixhauser_poa()` for the lookup table of the conditions which do an do not require associated ICD codes to be present-on-admission to flag the comorbidity.
- `get_icd_codes()` for the lookup table of all ICD codes.
- `get_pccc_codes()` for the lookup table of ICD codes used for the PCCC.
- `get_charlson_codes()` for the lookup table of ICD codes used for the Charlson comorbidities.
- `comorbidities()` for applying comorbidity algorithms to a data set.

Examples

```
head(get_elixhauser_codes())
str(get_elixhauser_codes())
```

---

get_elixhauser_index_scores

Get Elixhauser Index Scores

Description

Functions to get a copy of internal lookup tables for the ICD codes and index scores used in assessing Elixhauser comorbidities.

Usage

```
get_elixhauser_index_scores()
```

Value

A `data.frame` with the following columns:

- `condition`: Character vector of the conditions
- `index`: Character vector indicating if the score is for the mortality or the readmission index score
- `elixhauser_<variant>`: integer vector of the scores

See Also

- `get_elixhauser_codes()` for the lookup table of ICD codes mapping to the Elixhauser comorbidities.
- `get_elixhauser_poa()` for the lookup table of the conditions which do an do not require associated ICD codes to be present-on-admission to flag the comorbidity.
- `comorbidities()` for applying comorbidity algorithms to a data set.

Examples

```
head(get_elixhauser_index_scores())
str(get_elixhauser_index_scores())
```

`get_elixhauser_poa` *Get Elixhauser Present-on-Admission Requirements*

Description

Retrieve a copy of internal lookup table with details on which Elixhauser comorbidities do and do not require the associated ICD codes to be present-on-admission to be flagged.

Usage

```
get_elixhauser_poa()
```

Value

A `data.frame` with the following columns:

- `condition`: Character vector of the conditions
- `desc`: Character vector with a verbose description of the condition
- `poa_required`: Integer indicators if the code needs to present on admission to be considered a comorbidity
- `elixhauser_<variant>`: indicators for the Elixhauser `<variant>`

See Also

- `get_elixhauser_index_scores()` for the lookup table of the condition by condition scores for mortality and readmission indices.
- `get_elixhauser_codes()` for the lookup table of ICD codes mapping to the Elixhauser comorbidities.
- `comorbidities()` for applying comorbidity algorithms to a data set.

Examples

```
head(get_elixhauser_poa())
str(get_elixhauser_poa())
```

`get_icd_codes` *Get ICD Codes*

Description

Retrieve a copy of the internal lookup table for all known ICD codes.

Usage

```
get_icd_codes(with.descriptions = FALSE, with.hierarchy = FALSE)
```

Arguments

`with.descriptions`	Logical scalar, if `TRUE` include the description of the codes.
`with.hierarchy`	Logical scalar, if `TRUE` include the ICD hierarchy.

Details

Sources:

There are three sources of ICD codes.

- `cms`: Codes from the ICD-9-CM, ICD-9-PCS, ICD-10-CM, and ICD-10-PCS standards.
- `who`: Codes from World Health Organization.
- `cdc`: Codes from CDC Mortality coding standard.

Fiscal and Calendar Years:

When reporting years there is a mix of fiscal and calendar years.

Fiscal years are the United States Federal Government fiscal years, running from October 1 to September 30. For example, fiscal year 2013 started October 1 2012 and ended on September 30 2013.

Calendar years run January 1 to December 31.

Within the ICD data there are columns `known_start`, `known_end`, `assignable_start`, `assignable_end`, `desc_start` and `desc_end`. For ICD codes with `src == "cms"`, these are fiscal years. For codes with `src == "cdc"` or `src == "who"` these are calendar years.

`known_start` is the first fiscal or calendar year (depending on source) that the medicalcoder package as definitive source data for. ICD-9-CM started in the United States in fiscal year 1980. The CDC extracts included in medicalcoder span fiscal years 1997–2012; the CMS ICD-9-CM/PCS extracts start in fiscal year 2006 and run through fiscal year 2015. As such 1997 is the earliest "known start" for ICD-9 within medicalcoder.

`known_end` is the last fiscal or calendar year (depending on source) for which we have definitive source data for. For ICD-9-CM and ICD-9-PCS, CMS provides data through fiscal year 2015, while the CDC extracts stop at fiscal year 2012. For ICD-10-CM and ICD-10-PCS, which are active, it is just the last year of known data. ICD-10 from the WHO ends in 2019.

Header and Assignable Codes:

"Assignable" indicates that the code is the most granular for the source. Ideally codes are reported with the greatest level of detail but that is not always the case. Also, the greatest level of detail can differ between sources. Example: C86 is a header code for `cms` and `who` because codes C86.0, C86.1, C86.2, C86.3, C86.4, C86.5, and C86.6 all exist in both standards. No code with a fifth digit exists in the `who` so all these four digit codes are 'assignable.' In the `cms` standard, C86.0 was assignable through fiscal year 2024. In fiscal year 2025 codes C86.00 and C86.01 were added making C86.0 a header code and C86.00 and C86.01 assignable codes.

Value

a `data.frame`

The default return has the following columns:

- `icdv`: Integer vector indicating if the code is from ICD-9 or ICD-10
- `dx`: Integer vector. 1 if the code is a diagnostic, (ICD-9-CM, ICD-10-CM, WHO, CDC Mortality), or 0 if the code is procedural (ICD-9-PCS, ICD-10-PCS)
- `full_code`: Character vector with the ICD code and any relevant decimal point
- `code`: Character vector with the compact ICD code omitting any relevant decimal point
- `src`: Character vector reporting the source of the information. See Details.
- `known_start`: Integer vector reporting the first known year of use. See Details.
- `known_end`: Integer vector reporting the last known year of use. See Details.
- `assignable_start`: Integer vector reporting the first known year the code was assignable. See Details.
- `assignable_end`: Integer vector reporting the last known year the code was assignable. See Details.

When `with.descriptions = TRUE` there are the following additional columns:

- `desc`: Character vector of descriptions. For `cms` codes descriptions from CMS are used preferentially over CDC.
- `desc_start`: Integer vector of the first year the description was used.
- `desc_end`: Integer vector of the last year the description was used.

When `with.hierarchy = TRUE` there are the following additional columns:

- `chapter`
- `subchapter`
- `category`
- `subcategory`
- `subclassification`
- `subsubclassification`
- `extension`

See Also

is_icd(), lookup_icd_codes(), vignette(topic = "icd", package = "medicalcoder")

Examples

```
icd_codes <- get_icd_codes()
str(icd_codes)

# Explore the change in the assignable year for C86 code between CMS and
# WHO
subset(get_icd_codes(), grepl("^C86$", full_code))
subset(get_icd_codes(), grepl("^C86\\.\\d$", full_code))
subset(get_icd_codes(), grepl("^C86\\.0(\\d|$)", full_code))

is_icd("C86", headerok = FALSE) # FALSE
is_icd("C86", headerok = TRUE)  # TRUE
is_icd("C86", headerok = TRUE, src = "cdc") # Not a CDC mortality code

lookup_icd_codes("^C86\\.0\\d*", regex = TRUE)
```

---

get_pccc_codes *Pediatric Complex Chronic Conditions ICD Codes*

Description

Retrieve a copy of internal lookup tables for the ICD codes mapping to the Pediatric Complex Chronic Conditions (PCCC) conditions and subconditions by variant.

Usage

```
get_pccc_codes()
```

Value

a `data.frame` with the following columns

- `icdv`: Integer vector indicating if the code is from ICD-9 or ICD-10.
- `dx`: Integer vector. 1 if the code is a diagnostic, (ICD-9-CM, ICD-10-CM, WHO, CDC Mortality), or 0 if the code is procedural (ICD-9-PCS, ICD-10-PCS).
- `full_code`: Character vector with the ICD code and any relevant decimal point.
- `code`: Character vector with the compact ICD code omitting any relevant decimal point.
- `condition`: Character vector of the conditions.
- `subcondition`: Character vector of the subconditions.
- `transplant_flag`: Integer vector indicating if the code is associated with a transplant.
- `tech_dep_flag`: Integer vector indicating if the code is associated with technology dependence.
- `pccc_<variant>`: Integer vector indicating if the code is part of the v2.0, v2.1, v3.0, or v3.1 variant.

See Also

- `get_pccc_conditions()` for a reference of the PCCC conditions and subconditions.
- `get_icd_codes()` for the lookup table of all ICD codes.
- `comorbidities()` for applying comorbidity algorithms to a data set.

Examples

```
head(get_pccc_codes())
str(get_pccc_codes())
```

`get_pccc_conditions` *Pediatric Complex Chronic Condition and Subconditions*

Description

Retrieve a copy of internal lookup tables for the syntax valid and human readable labels of the Pediatric Complex Chronic Conditions (PCCC) conditions and subconditions.

Usage

```
get_pccc_conditions()
```

Value

a `data.frame` with the following columns

- `condition`: (character) syntax valid name for the condition
- `subcondition`: (character) syntax valid name for the subcondition
- `condition_label`: (character) human readable label for the condition
- `subcondition_label`: (character) human readable label for the subcondition

See Also

- `get_pccc_codes()` for the lookup table of ICD codes used for the PCCC.
- `comorbidities()` for applying comorbidity algorithms to a data set.

Examples

```
get_pccc_conditions()
```

`icd_compact_to_full` *Convert ICD Compact Codes to Full Codes*

Description

Take an assumed ICD compact code string and convert to a full code based on the ICD version (9 or 10) and type (diagnostic or procedure). This method only formats strings and does not validate the code(s).

Usage

```
icd_compact_to_full(x, icdv, dx)
```

Arguments

`x`	Character vector
`icdv`	Integer vector of allowed ICD versions. Use `9L` and/or `10L`. Defaults to both.
`dx`	Integer vector indicating allowed code type(s): `1L` for diagnostic (ICD-9-CM, ICD-10-CM, CDC mortality, WHO), `0L` for procedural (ICD-9-PCS, ICD-10-PCS). Defaults to both.

Value

A character vector the same length as `x`.

See Also

- `get_icd_codes()` to retrieve the internal lookup table of ICD codes.
- `lookup_icd_codes()` for retrieving details on a specific set of ICD codes.
- `is_icd()` to test if a string is a known ICD code.

Other ICD tools: `is_icd()`, `lookup_icd_codes()`

`is_icd` *Is ICD*

Description

Answer the question "is the character string `x` a valid ICD code?" ICD codes should be character vectors. `is_icd` will assess for both "full codes" (decimal point present when appropriate) and "compact codes" (decimal point omitted).

ICD-10 code "C00" is a header code because the four-character codes C00.0, C00.1, C00.2, C00.3, C00.4, C00.5, C00.6, C00.7, C00.8, and C00.9 exist. Those four-character codes are assignable (as of 2025) because no five-character descendants (e.g., C00.40) exist.

When the source is the World Health Organization (WHO) or CDC Mortality, years refer to calendar years. CDC/CMS sources use the U.S. federal fiscal year, which starts on October 1 (e.g., fiscal year 2024 runs 2023-10-01 to 2024-09-30).

Usage

```
is_icd(
  x,
  icdv = c(9L, 10L),
  dx = c(1L, 0L),
  src = c("cms", "who", "cdc"),
  year,
  headerok = FALSE,
  ever.assignable = missing(year),
  warn.ambiguous = TRUE,
  full.codes = TRUE,
  compact.codes = TRUE
)
```

Arguments

`x`	Character vector of ICD codes (full or compact form).
`icdv`	Integer vector of allowed ICD versions. Use `9L` and/or `10L`. Defaults to both.
`dx`	Integer vector indicating allowed code type(s): `1L` for diagnostic (ICD-9-CM, ICD-10-CM, CDC mortality, WHO), `0L` for procedural (ICD-9-PCS, ICD-10-PCS). Defaults to both.
`src`	Character vector of code sources. One or more of `"cms"`, `"who"`, `"cdc"`. Defaults to all.
`year`	Numeric scalar. Calendar or fiscal year to reference. Default is the most current year available per source. For ICD-9, CMS data run through fiscal year 2015 and CDC extracts through 2012; ICD-10 sources are updated annually. Calendar year for WHO and CDC mortality. Fiscal year for CMS.
`headerok`	Logical scalar. If `FALSE` (default), only assignable codes are considered valid; if `TRUE`, header codes are also accepted.
`ever.assignable`	Logical scalar. If `TRUE` then ignore `year` and return `TRUE` if the `x` was ever an assignable code.
`warn.ambiguous`	Logical scalar. If `TRUE` (default), warn when a code matches more than one ICD version and/or type (e.g., both CM and PCS).
`full.codes`	Logical scalar. If `TRUE` (default), match codes that include the decimal point where applicable.
`compact.codes`	Logical scalar. If `TRUE` (default), match codes without the decimal point.

Details

Similarly for ICD-9-CM: "055" is a header for measles; 055.0, 055.1, 055.2, 055.8, and 055.9 are assignable. Codes 055.3–055.6 do not exist. Code 055.7 is a header because 055.71 and 055.72 exist.

Some codes change status across years. For example, ICD-9-CM 516.3 was assignable in fiscal years 1997–2011 for the CDC extracts (2006–2011 for CMS) and became a header in 2012–2015.

Value

A logical vector the same length as `x`.

See Also

- `get_icd_codes()` to retrieve the internal lookup table of ICD codes.
- `lookup_icd_codes()` for retrieving details on a specific set of ICD codes.
- `icd_compact_to_full()` converts a string from a compact format to the full format based on ICD version and type (diagnostic or procedure).

Other ICD tools: `icd_compact_to_full()`, `lookup_icd_codes()`

Examples

```
################################################################################
# Some ICD-9 diagnostic codes
x <- c("136.2", "718.60", "642.02")

is_icd(x, icdv =  9, dx = 1)
is_icd(x, icdv =  9, dx = 0)
is_icd(x, icdv = 10, dx = 1)
is_icd(x, icdv = 10, dx = 0)

is_icd(x, icdv = 9, dx = 1, headerok = TRUE)
is_icd(x, icdv = 9, dx = 1, year = 2006)

################################################################################
# ICD code with, or without a dot.  The ICD-9 diagnostic code 799.3 and ICD-9
# procedure code 79.93 both become 7993 when assessed against the ICD code look
# up tables.  As such "7993" is a valid ICD-9 diagnostic and procedure code,
# whereas 799.3 is only a valid dx code, and 79.93 is only a valid pr code.
# Further, codes such as ".7993", "7.993", "7993.", are all non-valid codes.

x <- c("7993", ".7993", "7.993", "79.93", "799.3", "7993.")
data.frame(
  x,
  dx = is_icd(x, icdv = 9, dx = 1),
  pr = is_icd(x, icdv = 9, dx = 0)
)

################################################################################
# example of a ICD-9 code that was assignable, but became a header when
# more descriptive codes were introduced: ICD-9 diagnostic code 516.3
lookup_icd_codes(paste0("516.3", c("", as.character(0:9))))

# ICD-9 code 516.3 was an assignable code through fiscal year 2011.
is_icd("516.3")

# If `year` is omitted, and `ever.assignable = FALSE` then the `year` is
# implied to be the max `known_end` year for ICD codes matched by `icdv`,
# `dx`, and `src`.
is_icd("516.3", ever.assignable = FALSE)
```

```
# when `year` is provided then `ever.assignable` is `FALSE` by default and
# the return is TRUE when 516.3 was assignable and FALSE otherwise.
is_icd("516.3", year = 2015)
is_icd("516.3", year = 2011)

# when year is a non-assignable year, but `ever.assignable = TRUE` the return
# will be TRUE.  Useful if you know the data is retrospective and collected
# through fiscal year 2015.
is_icd("516.3", year = 2015, ever.assignable = TRUE)

################################################################################
# Consiser the string E010
#   - This could be a ICD-9-CM full code
#   - Could be a ICD-10-CM compact code
lookup_icd_codes("E010")
subset(get_icd_codes(with.descriptions = TRUE), grepl("^E010$", code))

is_icd("E010")
is_icd("E010", icdv = 9) # FALSE because it is a header code and was never assignable
is_icd("E010", icdv = 9, ever.assignable = TRUE) # FALSE
is_icd("E010", icdv = 9, headerok = TRUE) # TRUE
```

`lookup_icd_codes` *Lookup ICD Codes*

Description

Functions for working with ICD codes.

ICD-10 code "C00" is a header code because the four-character codes C00.0, C00.1, C00.2, C00.3, C00.4, C00.5, C00.6, C00.7, C00.8, and C00.9 exist. Those four-character codes are assignable (as of 2025) because no five-character descendants (e.g., C00.40) exist.

When the source is the World Health Organization (WHO) or CDC Mortality, years refer to calendar years. CDC/CMS sources use the U.S. federal fiscal year, which starts on October 1 (e.g., fiscal year 2024 runs 2023-10-01 to 2024-09-30).

Usage

```
lookup_icd_codes(
  x,
  regex = FALSE,
  full.codes = TRUE,
  compact.codes = TRUE,
  ...
)
```

Arguments

`x`	Character vector of ICD codes (full or compact form). When `regex = TRUE`, `x` must contain at least one non-empty, non-missing string.
`regex`	Logical scalar. If `TRUE`, treat `x` as regular expressions; if `FALSE`, use exact matching.
`full.codes`	Logical scalar. If `TRUE` (default), match codes that include the decimal point where applicable.
`compact.codes`	Logical scalar. If `TRUE` (default), match codes without the decimal point.
`...`	Passed to `base::grep()` when `regex = TRUE`

Details

ICD codes should be character vectors. These tools work with either "full codes" (decimal point present when appropriate) or "compact codes" (decimal point omitted).

Similarly for ICD-9-CM: "055" is a header for measles; 055.0, 055.1, 055.2, 055.8, and 055.9 are assignable. Codes 055.3–055.6 do not exist. Code 055.7 is a header because 055.71 and 055.72 exist.

Some codes change status across years. For example, ICD-9-CM 516.3 was assignable in fiscal years 1997–2011 for the CDC extracts (2006–2011 for CMS) and became a header in 2012–2015.

Value

A `data.frame` with one or more rows per input, including columns

- `match_type`: did the input match a full or compact code
- `icdv`: icd version (9 or 10)
- `dx`: diagnostic code (1) or procedure code (0)
- `full_code`: the full code string
- `code`: the compact codes string
- `src`: the source - CMS, CDC, or WHO.
- year ranges (`known_*`, `assignable_*`).

See Also

- `get_icd_codes()` to retrieve the internal lookup table of ICD codes.
- `is_icd()` to test if a string is a known ICD code.
- `icd_compact_to_full()` converts a string from a compact format to the full format based on ICD version and type (diagnostic or procedure).

Other ICD tools: `icd_compact_to_full()`, `is_icd()`

`mdcr` *Synthetic Data*

Description

Synthetic Data

Usage

```
mdcr
```

Format

`mdcr` is a `data.frame` with 4 columns, Each row is for one ICD id.

- `patid`: patient identifier, integer values
- `icdv`: ICD version; integer values, 9 or 10
- `dx`: indicator column for ICD diagnostic (1) or procedure (0) codes
- `code`: ICD code; character values

See Also

Other datasets: `mdcr_longitudinal`

`mdcr_longitudinal` *Synthetic Longitudinal Data*

Description

Synthetic Longitudinal Data

Usage

```
mdcr_longitudinal
```

Format

`mdcr_longitudinal` is a `data.frame` with 4 columns. The codes are expected to be treated as diagnostic codes. Warning: there are a few ICD-9 codes which could match to procedure codes.

- `patid`: patient identifier, integer values
- `date`: date the diagnostic code was recorded
- `icdv`: ICD version 9 or 10, integer valued
- `code`: ICD codes; character values

See Also

Other datasets: `mdcr`

summary.medicalcoder_comorbidities

Summaries of Comorbidities

Description

Build summaries (counts and percentages) for each comorbidity and other summary statistics by method.

Usage

```
## S3 method for class 'medicalcoder_comorbidities'
summary(object, ...)
```

Arguments

`object`	a `medicalcoder_comorbidities` object generated by calling `comorbidities()`
`...`	additional parameters, not currently used

Value

either a list or a data `data.frame`

Examples

```
pccc_v3.1_results <-
  comorbidities(data = mdcr,
                icd.codes = "code",
                id.vars = "patid",
                dx.var = "dx",
                method = "pccc_v3.1",
                flag.method = 'current',
                poa = 1)
summary(pccc_v3.1_results)

charlson_results <-
  comorbidities(data = mdcr,
                icd.codes = "code",
                id.vars = "patid",
                dx.var = "dx",
                method = "charlson_quan2011",
                flag.method = 'current',
                poa = 1)
summary(charlson_results)

elixhauser_results <-
  comorbidities(data = mdcr,
                icd.codes = "code",
                id.vars = "patid",
                dx.var = "dx",
                method = "elixhauser_ahrq2025",
                primarydx = 1,
                flag.method = 'current',
                poa = 1)
summary(elixhauser_results)
```

`summary.medicalcoder_comorbidities_with_subconditions`
Summaries of Comorbidities with Subconditions

Description

Build summaries (counts and percentages) for each Pediatric Complex Chronic Condition (PCCC) condition and subcondition.

Usage

```
## S3 method for class 'medicalcoder_comorbidities_with_subconditions'
summary(object, ...)
```

Arguments

`object`	a `medicalcoder_comorbidities_with_subconditions` object generated by calling `comorbidities()` with `subconditions = TRUE`. This is currently only applicable to PCCC.
`...`	additional parameters, not currently used

Value

a `data.frame` with five columns.

1. `condition` the primary condition
2. `subcondition` the subcondition(s) within the `condition`. There will be a row where `subcondition` is `NA` which is used to report the `count` and `percent_of_cohort` for the `condition` overall.
3. `count` the number of rows in `object` with the applicable `condition` and `subcondition`.
4. `percent_of_cohort`: a numeric value within [0, 100] for the percent of rows in `object` with the flagged `condition` and `subcondition`.
5. `percent_of_those_with_condition`: a numeric value within [0, 100] for the subset of rows in `object` with the primary `condition` and the flagged `subcondition`. Will be `NA` for the primary `condition`.

See Also

`comorbidities()`, `vignette(topic = "pccc", package = "medicalcoder")`

Examples

```
pccc_v3.1_subcondition_results <-
  comorbidities(data = mdcr,
                icd.codes = "code",
                id.vars = "patid",
                dx.var = "dx",
                method = "pccc_v3.1",
                flag.method = 'current',
                poa = 1,
                subconditions = TRUE)
summary(pccc_v3.1_subcondition_results)
```

Index