Package ‘utc2nwislocal’

June 8, 2020

Title UTC Date-Times to NWIS Local Time-Zone Codes using UTC Offsets

Version 1.0.1

Depends R (>= 3.0.0)

Date 2020-05-08

Author William H. Asquith [aut, cre], Ronald C. Seanor [aut]

Description U.S. Geological Survey National Water Information System (NWIS) time-zone code tables that are not fully congruent with those available natively in most R installations. Conversion of POSIXct date-times in Coordinated Universal Time (UTC) to a character string representation offset to NWIS time-zone codes is provided, and UTC offset computations for the NWIS time-zone codes are provided.

Maintainer William H. Asquith <wasquith@usgs.gov>

License CC0

Copyright This software is in the public domain because it contains materials that originally came from the United States Geological Survey, an agency of the United States Department of Interior. For more information, see the official USGS copyright policy at https://www.usgs.gov/information-policies-and-instructions/copyrights-and-credits

NeedsCompilation no

R topics documented:

`nwislocal2utc_offset_hours`
Convert a NWIS Local Time Zone to Offset in Hours from UTC

Description

Convert an U.S. Geological Survey National Water Information System (NWIS) (U.S. Geological Survey, 2019) time-zone code to the offset in hours from Coordinated Universal Time (UTC). The NWIS time-zone codes can be found at NWIS Time Zones (accessed on October 22, 2019) and see file `./inst/doc/README.md` in the **utc2nwislocal** sources for more details.

Usage

```
nwislocal2utc_offset_hours(tz)
```

Arguments

`tz`	A vector of NWIS time-zone codes (such as `lev_tz_cd`), and `NA` values are assumed to be in UTC with offset of zero;

Value

The UTC offsets in hours. Any time zone that does not exist in the lookup table (the hidden environment [hash table] `.NWIStzUTC$TimeZone_Offset`), is set to `NA` (see **Examples**).

Author(s)

W.H. Asquith

References

U.S. Geological Survey, 2019, USGS water data for the Nation: U.S. Geological Survey National Water Information System database, accessed August 9, 2019, at https://doi.org/10.5066/F7P55KJN.

See Also

`utc2nwislocal`, `utc_offset_hours2nwislocal`, `nwislocal2utc_offset_seconds`

Examples

```
nwislocal2utc_offset_hours(c("PST","PDT","the author",NA,"CDT"))
# Results, in hours, are: [1] -8 -7 NA  0 -5

# Force inspection of lookup table contents, using triple colon notation:
ls(utc2nwislocal:::.NWIStzUTC$TimeZone_Offset) # keys are time-zone codes
```

`nwislocal2utc_offset_seconds` *Convert a NWIS Local Time Zone to Offset in Seconds from UTC*

Description

Convert a U.S. Geological Survey National Water Information System (NWIS) (U.S. Geological Survey, 2019) time-zone code to the offset in seconds from Coordinated Universal Time (UTC). The NWIS time-zone codes can be found at NWIS Time Zones (accessed on October 22, 2019) and see file `./inst/doc/README.md` in the **utc2nwislocal** sources for more details.

Usage

```
nwislocal2utc_offset_seconds(tz)
```

Arguments

`tz` A vector of NWIS time-zone codes (such as `lev_tz_cd`), and `NA` values are assumed to be in UTC with offset of zero;

Value

The UTC offsets in seconds. Any time zone that does not exist in the lookup table (the hidden environment [hash table] `.NWIStzUTC$TimeZone_Offset_seconds`), is set to `NA` (see **Examples**).

Author(s)

W.H. Asquith

References

U.S. Geological Survey, 2019, USGS water data for the Nation: U.S. Geological Survey National Water Information System database, accessed August 9, 2019, at https://doi.org/10.5066/F7P55KJN.

See Also

`utc2nwislocal`, `nwislocal2utc_offset_hours`

Examples

```
nwislocal2utc_offset_seconds(c("PST","PDT","the author",NA,"CDT"))
# Results, in seconds, are: [1] -28800 -25200     NA      0 -18000

# Force inspection of lookup table contents, using triple colon notation:
ls(utc2nwislocal:::.NWIStzUTC$TimeZone_Offset_seconds) # keys are time-zone codes
```

`utc2nwislocal` *Convert UTC Date-Time to NWIS Local Date-Time*

Description

Convert Coordinated Universal Time (UTC) `base::as.POSIXct()` date-time values into various character-string representations for time zones. The UTC offset for individual time zones are determined from the globe-encompassing time-zone codes recognized by the U.S. Geological Survey National Water Information System (NWIS) (U.S. Geological Survey, 2019) (see **Note**). The time zones are tracked in a separate database slot than the UTC date-time.

The justification for this package is that the time-zone codes are not standard to the function `base::OlsonNames()` within base R, but the NWIS names are an ANSI SQL/92 time-zone offset string. American National Standards Institute (ANSI) SQL/92 was the third revision of the Structured Query Language (SQL) database query language. NWIS stores date-times exclusively in UTC, and certain data retrievals can be (purposefully) kept in UTC, though commonly NWIS or various data output methods (beyond the scope of this documentation) switches in output either to the user's local time zone or the USGS data-collection site, local time-zone code.

For some data operations external to NWIS, it is useful to retain UTC convention for the core date-time information and convert to local time on character-string output. Further, NWIS also retains a date-accuracy code, which is actively used to truncate character-string output of date-time values. The `utc2nwislocal` function as a result relies on three separate parts that are the first three arguments. The basis of this function is to add respectively UTC offsets (in seconds) to the UTC `as.POSIXct` date-times and convert the results into character strings.

Warning: The function for purposes of execution speed does not check that an UTC is assigned to the first argument `dt`.

The core operation of this function for the ith value is to add or subtract the number of seconds from the UTC:

```
  off <- NULL # reset for the try(), which traps unknown codes
  try(off <- get(tz[i], envir=.NWIStzUTC$TimeZone_Offset_seconds), silent=TRUE)
  if(is.null(off)) stop("fatal: ", off, " is an unrecognized NWIS TZ code")
 val <- dt[i] + off # the addition of seconds to UTC \code{as.POSIXct} date-time
```

and really the rest of the logic in the function is error trapping, adding optional features, and support for vectorization.

Usage

```
utc2nwislocal(dt, tz, acy=NA, no.ending.ws=FALSE, addat=FALSE,
                      nosec=FALSE, pad=TRUE, secpad=FALSE, tzpad=FALSE)
```

Arguments

`dt` A vector of `as.POSIXct` or similar date-time in UTC in which seconds can be subtracted or added in order to change the date-time. Also `NA` values for `dt` are returned as `NA`;

`tz` A vector of NWIS time-zone codes (such as `lev_tz_cd`) and `NA`s at which are assumed to be in UTC; thus it is the users responsibility to deliver UTC. This argument can be a factor, but internally, it is immediately flipped to a character representation. This argument also can be of length one, which will then be repeated to the length of `dt` internally;

`acy` The NWIS date-time accuracy code (such as `lev_dt_acy_cd`), which in implementation here is optional. This argument can be a factor, but internally, it is immediately flipped to a character representation. The `NA` values internally trigger *to-the-second* accuracy condition (though NWIS does not explicitly recognize an "s"econd condition *per se* because that is the default resolution of the database) and the entire date-time stamp of whatever accuracy held in `dt` is returned with the UTC offset applied. This argument also can be of length one, which will then be repeated to the length of `dt` internally;

`no.ending.ws` A logical to strip trailing whitespace from the end of the converted values. This is provided as switch to superceed the results of the various default right-padding styles;

`addat` A logical to trigger whether the "at" character is inserted instead of a space between the date and time component;

`nosec` A logical to override the `acy="s"` values by truncating the seconds out. This might be a more useful option than at first seen—consider that for vast majority of USGS datasets that seconds are simply `":00"` a consumer of this function might decide the truncate those to avoid unneeded character inflation in the returned string and subsequent impact on potential output files;

`pad` A logical to right-side pad a date-time stamp to 19 or 16 characters wide *before* the time-zone code is appended if it makes sense for the accuracy (codes $\in$ `Y`, `M`, `D`, `h`, and `m` but not `s`). This argument is intended to help the human eye more rapidly discern information from a wide variation in potential date-time stamps in output (see **Examples**);

`secpad` A logical controlling whether the date-time padding on the right-side is 19 (`secpad=TRUE`) or 16 (`secpad=FALSE`), and this option makes sense really only for `acy="m"`; and

`tzpad` A logical controlling whether the time-zone stamp is right-padded to 5 characters.

Value

A character vector of converted date and times from UTC to the time zones requested. The date-time stamp will be right-padded 16 or 19 characters wide, and the time-zone code is right-padded to 5 characters or not at all. This padding can be stripped by the `no.ending.ws` argument.

Note

A list of the globe-encompassing NWIS time-zone codes can be found at NWIS Time Zones (accessed on October 22, 2019) and the date-time accuracy codes can be found at NWIS Time Accuracy Codes (accessed on October 22, 2019). The conversion of time-zone code to UTC offset is made by tabular lookup. The `sysdata.rda` file (RData binary format) that comes with this package in `./R/sysdata.rda` contains the `environment` named `.NWIStzUTC`. Within this `environment` are three other environments titled
`TimeZone_Names`,
`TimeZone_Offset`, and
`TimeZone_Offset_seconds`.
The `.NWIStzUTC` and these other three are not exported by the **utc2nwislocal** package but they can be accessed, by curious users, through "triple-colon insistance (notation)." The following code shows how to access these three tables:

```
ls(utc2nwislocal:::.NWIStzUTC) # See the three table names
  # "TimeZone_Names"   "TimeZone_Offset"   "TimeZone_Offset_seconds"

  # The creation of a vector of all time-zone codes
all_tz_codes <- ls(utc2nwislocal:::.NWIStzUTC$TimeZone_Names)

  # Demonstration of access for a given code.
utc2nwislocal:::.NWIStzUTC$TimeZone_Names$IDLE
  #  "International Date Line, East"
utc2nwislocal:::.NWIStzUTC$TimeZone_Offset$IDLE
  # "+12:00"
utc2nwislocal:::.NWIStzUTC$TimeZone_Offset_seconds$IDLE
  #  43200
```

The code, which was used the create the `sysdata.rda`, is located in file `buildSYSDATA(R).txt` that is located in the `./inst/doc` subdirectory of this package's sources. NWIS appears to use a 5 (five) character wide (max) time-zone code convention. It is important to state that the table-lookup herein is based on white space being stripped from time-zone code. All spaces are internally stripped from the `tz` before processing commences—the pad is possible to return through the `tzpad` argument.

Author(s)

W.H. Asquith

References

U.S. Geological Survey, 2019, USGS water data for the Nation: U.S. Geological Survey National Water Information System database, accessed August 9, 2019, at https://doi.org/10.5066/F7P55KJN.

See Also

`nwislocal2utc_offset_hours`, `nwislocal2utc_offset_seconds`, `utc_offset_hours2nwislocal`

Examples

```
# Australia Eastern Standard Time is +10:00 to UTC and NWIS code == "AESST"
# EST, CDT, and PST are obviously American time zones.

dt_va <- as.POSIXct(c("2012-10-06 12:00:00", "2016-12-17 16:00:00",
                      "2017-07-26 06:15:00", "2017-07-26 06:15:30"), tz="UTC")

# Let us look only at impact on the Australian "data point."
utc2nwislocal(dt_va, tz="AESST")[1]      # tz vectored to length of dt_va
  # "2012-10-06 23:00:00 AESST"
utc2nwislocal(dt_va, tz="AESST", "h")[1] # m means accurate down to the hour
  # "2012-10-06 23     AESST"
utc2nwislocal(dt_va, tz="AESST", "s", nosec=TRUE)[1]
  # "2012-10-06 23:00  AESST"
utc2nwislocal(dt_va, tz="AESST", "M")[1]
  # "2012-10                "
utc2nwislocal(dt_va, tz="AESST", "Y")[1]
  # "2012                   "

utc2nwislocal(as.POSIXct("2011-03-14 15:30:00"), tz="MDT", "m", secpad=TRUE)
  # "2011-03-14 09:30    MDT"
utc2nwislocal(as.POSIXct("2011-03-14 15:30:00"), tz="MDT", "m", secpad=FALSE)
  # "2011-03-14 09:30 MDT"

tzs <- c("AESST", "EST", "CDT", "PST")
utc2nwislocal(dt_va, tz=tzs)      # tz vectored to length of dt_va
utc2nwislocal(dt_va, tz=tzs, "h") # hour means accurate down to the hour
utc2nwislocal(dt_va, tz=tzs, "s", nosec=TRUE)
utc2nwislocal(dt_va, tz=tzs, c("D","h","m","s"))
```

`utc_offset_hours2nwislocal` *Convert a Offset in Hours from UTC to NWIS Local Time Zone*

Description

Convert an offset in hours from Coordinated Universal Time (UTC) to a U.S. Geological Survey National Water Information System (NWIS) (U.S. Geological Survey, 2019) time-zone code. The NWIS time-zone codes can be found at NWIS Time Zones (accessed on October 22, 2019) and see file `./inst/doc/README.md` in the **utc2nwislocal** sources for more details.

Usage

```
utc_offset_hours2nwislocal(hrs, split=FALSE, trace=FALSE)
```

Arguments

hrs — A vector of NWIS time-zone codes (such as lev_tz_cd), and NA values are assumed to be in UTC with offset of zero;

split — A logical triggering a unlist(strsplit()) on the time-zone codes (see **Examples**); and

trace — A logical triggering a message() where the user can see the looping through the time-zone codes.

Value

One or more time-zone codes matching the hrs (if any) as affected by the split argument. The string " and " is inserted between the time-zone codes if split=FALSE; otherwise, a vector of the time-zone-codes results. Any time zone that does not exist in the lookup table (the hidden environment [hash table] .NWIStzUTC$TimeZone_Offset), is set to NA (see **Examples**).

Author(s)

W.H. Asquith

References

U.S. Geological Survey, 2019, USGS water data for the Nation: U.S. Geological Survey National Water Information System database, accessed August 9, 2019, at https://doi.org/10.5066/F7P55KJN.

See Also

nwislocal2utc_offset_hours

Examples

```
utc_offset_hours2nwislocal("+02:00", split=FALSE)
# [1] "CETDST and EET and FWT and IST and MEST and METDST and SST"

utc_offset_hours2nwislocal("+02:00", split=TRUE)
# [1] "CETDST" "EET"    "FWT"    "IST"    "MEST"   "METDST" "SST"

utc_offset_hours2nwislocal( "+2:00", split=FALSE) # must have proper syntax
# [1] NA
```

Index