

# Package: lutz (via r-universe)

September 10, 2024

**Type** Package

**Title** Look Up Time Zones of Point Coordinates

**Version** 0.3.2.9000

**Description** Input latitude and longitude values or an 'sf/sfc' POINT object and get back the time zone in which they exist. Two methods are implemented. One is very fast and uses 'Rcpp' in conjunction with data from the 'Javascript' library (<https://github.com/darkskyapp/tz-lookup-oss/>). This method also works outside of countries' borders and in international waters, however speed comes at the cost of accuracy - near time zone borders away from populated centres there is a chance that it will return the incorrect time zone. The other method is slower but more accurate - it uses the 'sf' package to intersect points with a detailed map of time zones from here: <https://github.com/evansiroky/timezone-boundary-builder/>. The package also contains several utility functions for helping to understand and visualize time zones, such as listing of world time zones, including information about daylight savings times and their offsets from UTC. You can also plot a time zone to visualize the UTC offset over a year and when daylight savings times are in effect.

**License** MIT + file LICENSE

**URL** <https://andyteacher.ca/lutz/>, <https://github.com/ateucher/lutz>

**BugReports** <https://github.com/ateucher/lutz/issues>

**Depends** R (>= 3.2)

**Imports** stats, Rcpp, lubridate

**Suggests** testthat (>= 2.1.0), sf (>= 0.7), sp, datasets, covr, ggplot2

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.2.3

**Roxygen** list(markdown = TRUE)

**LinkingTo** Rcpp

**Repository** <https://ateucher.r-universe.dev>

**RemoteUrl** <https://github.com/ateucher/lutz>

**RemoteRef** HEAD

**RemoteSha** bddb200863e54e4d04752b2aef03f4f5c6929fc

## Contents

|                            |          |
|----------------------------|----------|
| tz_list . . . . .          | 2        |
| tz_lookup . . . . .        | 3        |
| tz_lookup_coords . . . . . | 4        |
| tz_offset . . . . .        | 5        |
| tz_plot . . . . .          | 5        |
| <b>Index</b>               | <b>6</b> |

---

|         |                                    |
|---------|------------------------------------|
| tz_list | <i>Create a list of Time Zones</i> |
|---------|------------------------------------|

---

### Description

Output a list of time zone names, with daylight savings time and utc offset

### Usage

```
tz_list()
```

### Value

A data.frame of all time zones on your system. Columns:

- tz\_name: the name of the time zone
- zone: time zone
- is\_dst: is the time zone in daylight savings time
- utc\_offset\_h: offset from UTC (in hours)

---

|           |   |
|-----------|---|
| tz_lookup | <i>Lookup time zones of sf or sp points</i> |
|-----------|---|

---

### Description

There are two methods - "fast", and "accurate". The "fast" version can look up many thousands of points very quickly, however when a point is near a time zone boundary and not near a populated centre, it may return the incorrect time zone. If accuracy is more important than speed, use method = "accurate".

### Usage

```
tz_lookup(x, crs = NULL, method = "fast", warn = TRUE)
```

### Arguments

|        |  |
|--------|--|
| x      | either an sfc or sf points or SpatialPoints(DataFrame) object  |
| crs    | the coordinate reference system: integer with the EPSG code, or character with proj4string. If not specified (i.e., NULL) and x has no existing crs, EPSG: 4326 is assumed (lat/long). |
| method | method by which to do the lookup. Either "fast" (default) or "accurate".   |
| warn   | By default, if method = "fast" a warning is issued about the potential for inaccurate results. Set warn to FALSE to turn this off.   |

### Details

Note that there are some regions in the world where a single point can land in two different overlapping time zones. The "accurate" method includes these, and when they are encountered they are concatenated in a single string, separated by a semicolon. The data used in the "fast" method does not include overlapping time zones at this time.

### Value

character vector the same length as x specifying the time zone of the points.

### Examples

```
if (require("sf")) {
  state_pts <- lapply(seq_along(state.center$x), function(i) {
    st_point(c(state.center$x[i], state.center$y[i]))
  })
  state_centers_sf <- st_sf(st_sfc(state_pts))
  state_centers_sf$tz <- tz_lookup(state_centers_sf)
```

```
plot(state_centers_sf[, "tz"])
}
```

---

tz\_lookup\_coords      *Lookup time zones of lat/long pairs*

---

### Description

There are two methods - "fast", and "accurate". The "fast" version can look up many thousands of points very quickly, however when a point is near a time zone boundary and not near a populated centre, it may return the incorrect time zone. If accuracy is more important than speed, use method = "accurate".

### Usage

```
tz_lookup_coords(lat, lon, method = "fast", warn = TRUE)
```

### Arguments

|        |  |
|--------|--|
| lat    | numeric vector of latitudes  |
| lon    | numeric vector of longitudes the same length as x  |
| method | method by which to do the lookup. Either "fast" (default) or "accurate".   |
| warn   | By default, if method = "fast" a warning is issued about the potential for inaccurate results. Set warn to FALSE to turn this off. |

### Value

character vector the same length as x and y specifying the time zone of the points.

### Examples

```
tz_lookup_coords(42, -123)
tz_lookup_coords(lat = c(48.9, 38.5, 63.1, -25), lon = c(-123.5, -110.2, -95.0, 130))
```

---

|           |   |
|-----------|---|
| tz_offset | <i>Find the offset from UTC at a particular date/time in a particular time zone</i> |
|-----------|---|

---

**Description**

Find the offset from UTC at a particular date/time in a particular time zone

**Usage**

```
tz_offset(dt, tz = "")
```

**Arguments**

|    |   |
|----|---|
| dt | Date, POSIXt or date-like character string  |
| tz | A time zone name from <code>base::OlsonNames()</code> . Not required if dt is a POSIXt object with a time zone component. |

**Value**

a one-row data frame with details of the time zone

**Examples**

```
tz_offset("2018-06-12", "America/Moncton")
```

---

|         |                         |
|---------|-------------------------|
| tz_plot | <i>Plot a time zone</i> |
|---------|-------------------------|

---

**Description**

Make a circular plot of a time zone, visualizing the UTC offset over the course of the year, including Daylight Savings times

**Usage**

```
tz_plot(tz)
```

**Arguments**

|    |   |
|----|---|
| tz | a valid time zone name. See <code>OlsonNames()</code> |
|----|---|

**Value**

a ggplot2 object

**Examples**

```
tz_plot("America/Vancouver")
```

# Index

`base::OlsonNames()`, [5](#)

`OlsonNames()`, [5](#)

`tz_list`, [2](#)

`tz_lookup`, [3](#)

`tz_lookup_coords`, [4](#)

`tz_offset`, [5](#)

`tz_plot`, [5](#)