Time conversion and regularization

tsbox relies on a set of converters to convert time series stored as **ts**, **xts**, **data.frame**, **data.table**, **tibble**, **zoo**, **tsibble**, **tibbletime** or **timeSeries** to each other. This vignette describes some background on two particular challenges, the conversion of *equispaced points in time* to *actual dates or times*, and the *regularization of irregular time sequences*.

The classic way of storing time series in R in `"ts"` objects. These are simple vectors with an attribute that describes the beginning of the series, the (redundant) end, and the frequency. Thus, a monthly series, e.g., `AirPassengers`, is defined as a numeric vector that starts in 1949, with frequency 1. Thus, months are thought of as equispaced periods with a length of exactly 1/12 of a year.

For most time series, this is not what is meant. The second period of `AirPassengers`, February 1949, is actually shorter than the first one, but this is not reflected in the `"ts"` object. When converting to classes with actual time stamps, tsbox tries to correct it by using **heuristic**, rather than **exact** time conversion if possible.

Heuristic time conversion

Whenever possible, tsbox relies on **heuristic time conversion**. When a monthly `"ts"` time series, e.g., `AirPassengers`, is converted to a data frame, each time stamp (of class `"Date"`) indicates the first day of the month. Heuristic conversion is done for the following frequencies:

`ts`-frequency	time difference
365.2425	1 day
12	1 month
6	2 month
4	3 month
3	4 month
2	6 month
1	1 year
0.5	2 year
0.333	3 year
0.25	4 year
0.2	5 year
0.1	10 year

For example, converting `AirPassengers` to a data frame returns:

```
head(ts_df(AirPassengers))
#>         time value
#> 1 1949-01-01   112
```

```
#> 2 1949-02-01   118
#> 3 1949-03-01   132
#> 4 1949-04-01   129
#> 5 1949-05-01   121
#> 6 1949-06-01   135
```

Heuristic conversion works both ways, so we can get back to the original `"ts"` object:

```
all.equal(ts_ts(ts_df(AirPassengers)), AirPassengers)
#> [1] TRUE
```

Exact time conversion

For non standard frequencies, e.g. 260, of `EuStockMarkets`, tsbox uses **exact time conversion**. The year is divided into 260 equispaced units, each somewhat longer than a day. The time stamp of a period will be an exact point in time (of class `"POSIXct"`).

```
head(ts_df(EuStockMarkets))
#>    id                time   value
#> 1 DAX 1991-07-01 03:18:27 1628.75
#> 2 DAX 1991-07-02 13:01:32 1613.63
#> 3 DAX 1991-07-03 22:44:38 1606.51
#> 4 DAX 1991-07-05 08:27:43 1621.04
#> 5 DAX 1991-07-06 18:10:48 1618.16
#> 6 DAX 1991-07-08 03:53:53 1610.61
```

Higher frequencies, such as days, hours, minutes or seconds, are naturally equispaced, and exact time conversion is used as well.

Exact time conversion is generally reversible:

```
all.equal(ts_ts(ts_df(EuStockMarkets)), EuStockMarkets)
#> [1] TRUE
```

However, for high frequencies, rounding errors can lead to unavoidable small differences when going from data frame to `"ts"` and back.

Conversion does not work reliably if the frequency higher than one second. For these ultra high frequencies, tsbox is not tested and may not work as expected.

Regularization

In data frames or `"xts"` objects, missing values are generally omitted. These omitted missing values are called implicit, as opposite to explicit `NA` values. The function `ts_regular` allows the user to *regularize* a series, by making implicit missing values explicit.

When regularizing, `ts_regular` analyzes the differences in the time stamp for known frequencies. If it detects any, it builds a regular sequence based on the highest known frequency, and tries to match the

time stamps to the regular series. The result is a data frame or `"xts"` object with explicit missing values. Regularization is automatically done when an object is converted to a `"ts"` object.

For example, the following time series contains an implicit `NA` value in February 1974:

```
df <- ts_df(fdeaths)[-2,]
head(df)
#>         time value
#> 1 1974-01-01   901
#> 3 1974-03-01   827
#> 4 1974-04-01   677
#> 5 1974-05-01   522
#> 6 1974-06-01   406
#> 7 1974-07-01   441
```

`ts_regular` can be used to turn it into a explicit `NA`:

```
head(ts_regular(df))
#>         time value
#> 1 1974-01-01   901
#> 2 1974-02-01    NA
#> 3 1974-03-01   827
#> 4 1974-04-01   677
#> 5 1974-05-01   522
#> 6 1974-06-01   406
```

Regularization can be done for all frequencies that are suited for heuristic conversion, as listed above. In addition to these frequencies, the following higher frequencies are detected and regularized as well:

`ts`-frequency	time difference
31556952	1 sec
15778476	2 sec
6311390	5 sec
3155695	10 sec
2103797	15 sec
1577848	20 sec
1051898	30 sec
525949.2	1 min
262974.6	2 min
105189.8	5 min
52594.92	10 min
35063.28	15 min
26297.46	20 min

ts-frequency	time difference
17531.64	30 min
8765.82	1 hour
4382.91	2 hour
2921.94	3 hour
2191.455	4 hour
1460.97	6 hour
730.485	12 hour
365.2425	1 day