DSC 534–Lab 3

Yiolanda Englezou

The ts class in R

For defining a time series of class ts, we need to

- provide the data
- the starting time as argument *start*,
- the frequency of measurements as argument *frequency*.

If no starting time is supplied, R uses its default value of 1, i.e. enumerates the times by $1, \ldots, n$, where n is the length of the series.

The frequency is the number of observations per unit of time, e.g. 1 for yearly, 4 for quarterly, or 12 for monthly recordings. Instead of the start, we could also provide the end of the series, and instead of the frequency, we could supply argument deltat, the fraction of the sampling period between successive observations. The following example (hypotehtical data) illustrates these concepts

```
set.seed(123456)
raw.dat  <- rpois(10, 50)   ### yeary number of  HIV cases in Cyprus 2014--2023
ts.dat <- ts(raw.dat, start=2014, freq=1)
ts.dat
```

```
## Time Series:
## Start = 2014
## End = 2023
## Frequency = 1
##  [1] 55 48 47 40 55 59 67 58 46 40
```

```
##obtain useful information
start(ts.dat)
```

```
## [1] 2014    1
```

```
end(ts.dat)
```

```
## [1] 2023    1
```

```
frequency(ts.dat)
```

```
## [1] 1
```

```
deltat(ts.dat)
```

```
## [1] 1
```

Some other uses

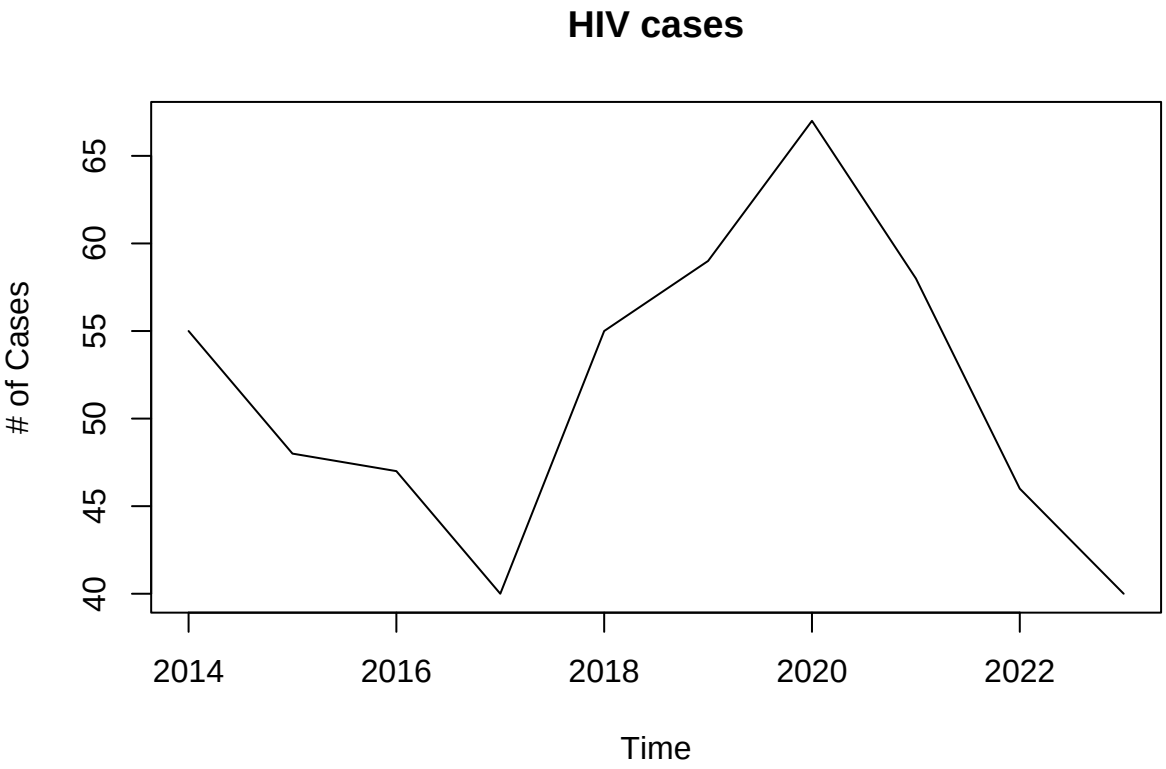
```
#obtain the measurement times from a time series object. They are given as fractions.
#Useful for specialized plots, etc.
time(ts.dat)
```

```
## Time Series:
## Start = 2014
## End = 2023
## Frequency = 1
##  [1] 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023
```

```
#selecting a subset from a time series. Regular R-subsetting still  works with the time series cla
window(ts.dat, start=2016, end=2019)
```

```
## Time Series:
## Start = 2016
## End = 2019
## Frequency = 1
## [1] 47 40 55 59
```

```
###plot the data
plot(ts.dat, ylab="# of Cases", main="HIV cases")
```



Dates and Time in R

While for the ts class, the handling of times has been solved very simply and easily by enumerating, doing time series analysis in R may sometimes also require to explicitly working with date and time.

- The built-in as.Date() function handles dates that come without times.
- The contributed package chron handles dates and times, but does not take into account different time zones
- POSIXct and POSIXlt classes allow for dates and times with time zone control. (compliacted)

```
Sys.setlocale("LC_TIME","en_US")
```

```
## [1] "en_US"
```

```r
as.Date("2024-08-31")
```

```
## [1] "2024-08-31"
```

```r
as.Date("2024/08/31")
```

```
## [1] "2024-08-31"
```

```r
#Date objects are stored as the number of days passed since the 1st of
#January in 1970. Earlier dates receive negative numbers. Use the
#as.numeric() function, to find out how many days are past since the
#reference date. Also back-conversion from a number of past days to a date is easily implemented
mydat <- as.Date("2023-10-14")
ndays <- as.numeric(mydat)
ndays
```

```
## [1] 19644
```

```r
tdays <- 10000
class(tdays) <- "Date"
tdays
```

```
## [1] "1997-05-19"
```

```r
#Extracting weekdays, months and quarters
#from Date objects and this information can be converted to
#factors. In this form, we can use them for visualization, decomposition,
#or time series regression.
weekdays(mydat)
```

```
## [1] "Saturday"
```

```r
months(mydat)
```

```
## [1] "October"
```

```r
quarters(mydat)
```

```
## [1] "Q4"
```

```r
###Summary statistics

dat <- as.Date(c("2010-01-01","2024-04-04","2017-08-09"))
min(dat)
```

```
## [1] "2010-01-01"
```

```r
max(dat)
```

```
## [1] "2024-04-04"
```

```r
mean(dat)
```

```
## [1] "2017-04-15"
```

```r
median(dat)
```

```
## [1] "2017-08-09"
```

```r
dat[3]-dat[1]
```

```
## Time difference of 2777 days
```

```r
##generate sequence of dates
seq(as.Date("2024-01-01"), by="days", length=15)
```

```
##  [1] "2024-01-01" "2024-01-02" "2024-01-03" "2024-01-04" "2024-01-05"
##  [6] "2024-01-06" "2024-01-07" "2024-01-08" "2024-01-09" "2024-01-10"
## [11] "2024-01-11" "2024-01-12" "2024-01-13" "2024-01-14" "2024-01-15"
```

```r
###using the by argument
seq(as.Date("2024-01-01"), by="2 weeks", length=12)
```

```
##  [1] "2024-01-01" "2024-01-15" "2024-01-29" "2024-02-12" "2024-02-26"
##  [6] "2024-03-11" "2024-03-25" "2024-04-08" "2024-04-22" "2024-05-06"
## [11] "2024-05-20" "2024-06-03"
```