# Dates and date-times with lubridate

Brendan Clarke, NHS Education for Scotland, brendan.clarke2@nhs.scot

17/05/2024

KIND Learning Network

# WoW

- dates and times are hard everywhere
- R is no exception
- this session is a beginner's guide to lubridate
  - not the only way of dealing with dates
  - not always the best
  - on balance the most consistent, and least quirky tools for dates

# This session

- beginner-friendly
- focus on core parsing, get/set, and rounding functions
- lots on dates, a bit of date-times, no times

## Resources

```
1 #install.packages("lubridate")
2 library(lubridate)
```

- Lubridate cheatsheet
- R4DS 2e chapter on dates and times
- quick primer on ISO8601

KIND Learning Network
NHS Education for Scotland

# R dates

- **days** since 1970-01-01

```
1  as_date(0)
```

[1] “1970-01-01”

# R dates

```
1 as_date(19860)
```

[1] “2024-05-17”

```
1 as_date(1:5)
```

[1] “1970-01-02” “1970-01-03” “1970-01-04” “1970-01-05” “1970-01-06”

```
1 class(as_date(0))
```

[1] “Date”

# R date-times

- **seconds** since 1970-01-01 00:00:00 UTC

```
1 as_datetime(0)
```

[1] “1970-01-01 UTC”

```
1 as_datetime(1715935369)
```

[1] “2024-05-17 08:42:49 UTC”

```
1 as_datetime(0:5)
```

[1] “1970-01-01 00:00:00 UTC” “1970-01-01 00:00:01 UTC” [3] “1970-01-01 00:00:02 UTC” “1970-01-01 00:00:03 UTC” [5] “1970-01-01 00:00:04 UTC” “1970-01-01 00:00:05 UTC”

```
1 class(as_datetime(0))
```

[1] “POSIXct” “POSIXt”

# Famously...

```
1  as_datetime(2 ^ 31-1) # 32 bit signed int
```

[1] “2038-01-19 03:14:07 UTC”

# Parsing dates is important

- most functions that accept dates (like ggplot) will mis-behave if you feed them date-shaped-words
  - e.g. alphabetically-ordered dates
- we also want to be able to calculate with dates

# Couple of fun intro functions

```
1 today()
```

[1] “2024-05-17”

```
1 date_decimal(2024.37534)
```

[1] “2024-05-17 08:59:11 UTC”

```
1 now()
```

[1] “2024-05-17 09:27:08 BST”

```
1 now("Japan")
```

[1] “2024-05-17 17:27:08 JST”

```
1 random_zone <- sample(OlsonNames(), 1)
2 cat(paste("The date-time in", random_zone, "is", now(sample(OlsonNames(), 1))))
```

The date-time in Asia/Baku is 2024-05-17 09:27:09.060887

# Parsing

- `as_date()` is fine assuming you have your date as a number of days
- but usually, we'll need to **parse** our dates

```
1 as_date(45429) # excel-format 1900 date
```

[1] "2094-05-19"

```
1 as_date(45429 - 25569) # dirty but effective
```

[1] "2024-05-17"

```
1 as_date(45429, origin = "1899-12-30") # better
```

[1] "2024-05-17"

# Parsing

- more often, we'll be taking human-readable dates and parsing them
- that's a pain, because there are loads of inconsistent ways of representing dates
- worse, lots of dates are ambiguous (5/6/24 and 6/5/24 might refer to the same day)

```
1 date_input <- c("17/5/24", "2024-05-17", "Friday 17th May 2024", "17*May*24", "5/17/2024" )
2
3 as_date(date_input) # as_date expects ISO-8601ish dates
```

[1] "2017-05-24" "2024-05-17" NA "2017-05-24" NA

```
1 # one correct, two silently incorrent, two NAs
```

KIND Learning Network

# parse_date_time

```
1 parse_date_time(date_input, orders = "ymd")
```

[1] “2017-05-24 UTC” “2024-05-17 UTC” NA “2017-05-24 UTC” [5] NA

```
1 parse_date_time(date_input, orders = c("dmy", "ymd", "dmy", "dmy", "mdy"))
```

[1] “2024-05-17 UTC” “2024-05-17 UTC” “2024-05-17 UTC” “2024-05-17 UTC” [5] “2024-05-17 UTC”

# dmy and co

- you can also use the orders (like dmy) as standalone parsing functions:

```
1  dmy(date_input[c(1,3,4)])
```

[1] “2024-05-17” “2024-05-17” “2024-05-17”

```
1  ymd_hms("2024-05/17 9-05-01")
```

[1] “2024-05-17 09:05:01 UTC”

# So you can make dates/date-times. So what?

```
1 date(now())
```

[1] “2024-05-17”

```
1 year(today())
```

[1] 2024

```
1 leap_year(today())
```

[1] TRUE

```
1 quarter(today())
```

[1] 2

```
1 semester(today())
```

[1] 1

```
1 semester(today(), with_year = T)
```

KIND Learning Network

NHS Education for Scotland

[1] 2024.1

KIND
Learning Network
NHS Education for Scotland

# Months and weeks

```
1 month(today())
```

[1] 5

```
1 month(today(), label = T)
```

[1] May 12 Levels: Jan < Feb < Mar < Apr < May < Jun < Jul < Aug < Sep < … < Dec

```
1 week(today())
```

[1] 20

```
1 epiweek(today()) # special ways of counting weeks. See  https://en.wikipedia.org/wiki/ISO_week_date and http
```

[1] 20

# Days

```
1 day(today())
```

[1] 17

```
1 wday(today())
```

[1] 6

```
1 qday(today())
```

[1] 47

# Hour and minute

```
1 hour(now())
```

[1] 9

```
1 minute(now())
```

[1] 27

```
1 am(now())
```

[1] TRUE

```
1 dst(now())
```

[1] TRUE

# Set

```
update(now(), hour = 11, minute = 0, second = 0) # nominal finish time today
```

[1] “2024-05-17 11:00:00 BST”

- or, more generally:

```
test_date <- dmy("05/06/23")
day(test_date)
```

[1] 5

```
day(test_date) <- 11
test_date
```

[1] “2023-06-11”

# Round

```
1 floor_date(today(), unit = "week")
```

[1] “2024-05-12”

```
1 round_date(today(), unit = "week")
```

[1] “2024-05-19”

```
1 ceiling_date(today(), unit = "month")
```

[1] “2024-06-01”

```
1 rollback(today()) # last day of previous month
```

[1] “2024-04-30”