Working with dates using `lubridate`

Emily Malcolm-White

Dates and times are everywhere in data: timestamps on social media posts, transaction dates in sales records, birthdates in survey data. But unlike numbers or strings, dates are tricky—you can't just subtract them like normal numbers, and parsing them from messy formats can be a headache.

Date Formats

Think of how many different formats you know of to format a date:

- 2023 07 06
- Wed, Jun 7, 2023
- 07-06-23
- 06-07-23 14:55 ET
- 06/07/2023 2:55pm

Yikes!

Date, Time, and Datetime

Date/time data are data that conveys information about, you guessed it, date and/or time! There are three relevant data types when we talk about date/time data:

1. Date - only has the date (e.g. 2020-05-15)
2. Time - only has the time (e.g. 20:45:00)
3. Datetime - has both the date and time (e.g. 2020-05-15 20:45:00)

Lubridate

Figure 1: Artwork by @allisonhorst

```
#LOAD PACKAGES
library(tidyverse)
library(lubridate)
```

Standard Date Format

The `ymd()` function transforms data in all kinds of different formats into a standardized date format displaying year, then month, then day.

```
ymd("06 02 04")
```

```
[1] "2006-02-04"
```

```
ymd("06/02/04")
```

```
[1] "2006-02-04"
```

```
ymd("20060204")  # works as well
```

```
[1] "2006-02-04"
```

```
ymd("2006 2 4")
```

```
[1] "2006-02-04"
```

```
ymd(060204)  # works with numbers
```

```
[1] "2006-02-04"
```

`mdy()` (month day year) and `dmy()` (day month year) formats also exist.

```
ymd_hms("2020-04-01 10:30:13")
```

```
[1] "2020-04-01 10:30:13 UTC"
```

```
ymd_hm("2020/04/01 10.30")
```

```
[1] "2020-04-01 10:30:00 UTC"
```

Extracting Components

Figure 2: Artwork by @allisonhorst

Once you have a date object, you can easily extract parts of it:

```r
birthday <- ymd("1998-09-27")

year(birthday)      # 1998
```

```
[1] 1998
```

```r
month(birthday)     # 9
```

```
[1] 9
```

```r
month(birthday, label = TRUE)  # "Sep"
```

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

```r
day(birthday)       # 27
```

```
[1] 27
```

```r
wday(birthday, label = TRUE)   # "Sun"
```

```
[1] Sun
Levels: Sun < Mon < Tue < Wed < Thu < Fri < Sat
```

Doing Math with Dates

Set up the object `today` as today's date:

```r
today <- Sys.Date()
```

Calculate the age based on today's date and your birthday:

```r
age <- today - birthday
age
```

```
Time difference of 9688 days
```

```
time_length(age, "years" )
```

```
[1] 26.5243
```

What is will be the date 28 days from now?

```
today + days(28)
```

```
[1] "2025-05-04"
```

Some real life examples:

Recall the Portal Project – a long-term ecological study being conducted near Portal, AZ. Since 1977, the site has been used to study the interactions among rodents, ants and plants and their respective responses to climate.

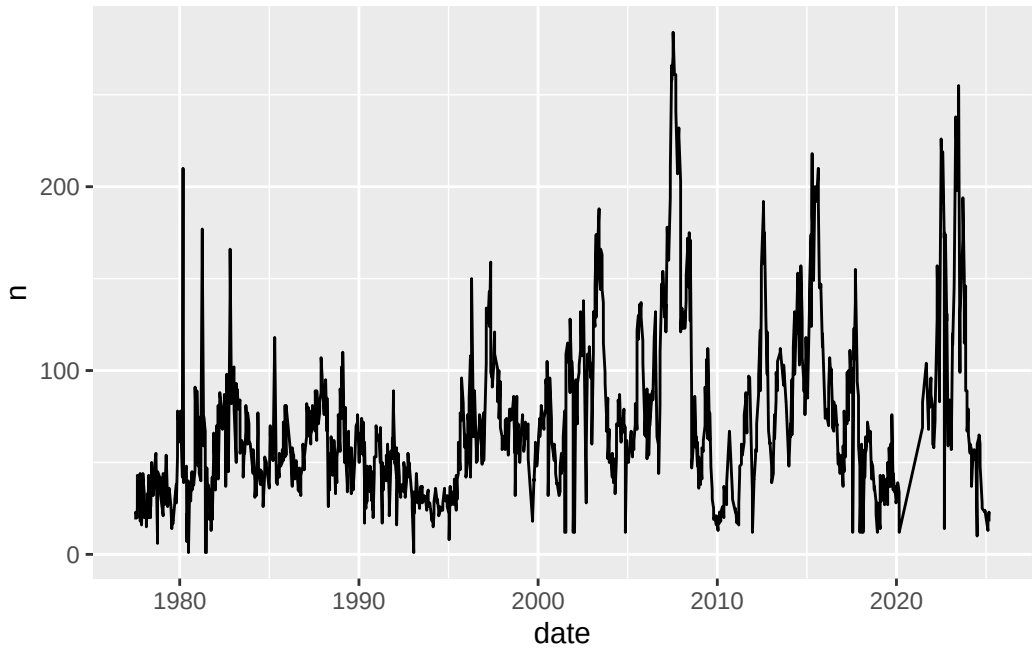
```
#LOAD DATA
portal_rodent <- read.csv("https://github.com/weecology/PortalData/raw/main/Rodents/Portal
```

Unfortunately, because the information about datetime is divided up into different columns, R does not recognize it as date/time data. What we need to do is combine and convert all of these columns into datetime. To do this, we can use the function `make_datetime()`

```
portal_rodent <- portal_rodent %>%
  mutate(date = make_date(year, month, day))
```

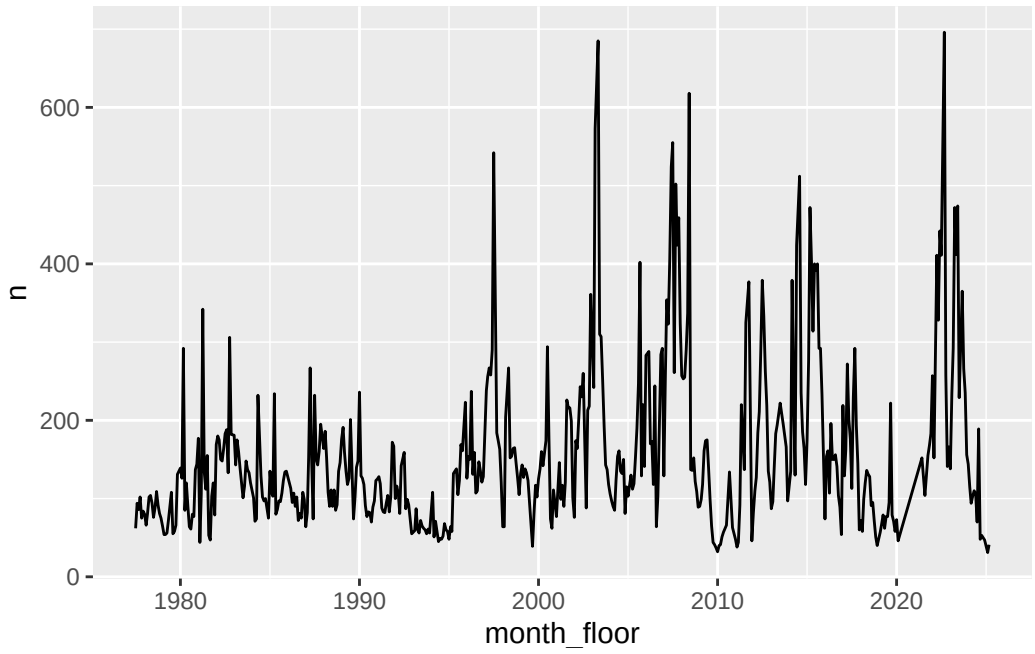
A line plot showing the number of rodents captured per day over time:

```
portal_rodent %>%
  count(date) %>%
  ggplot(aes(x = date, y = n)) +
  geom_line()
```



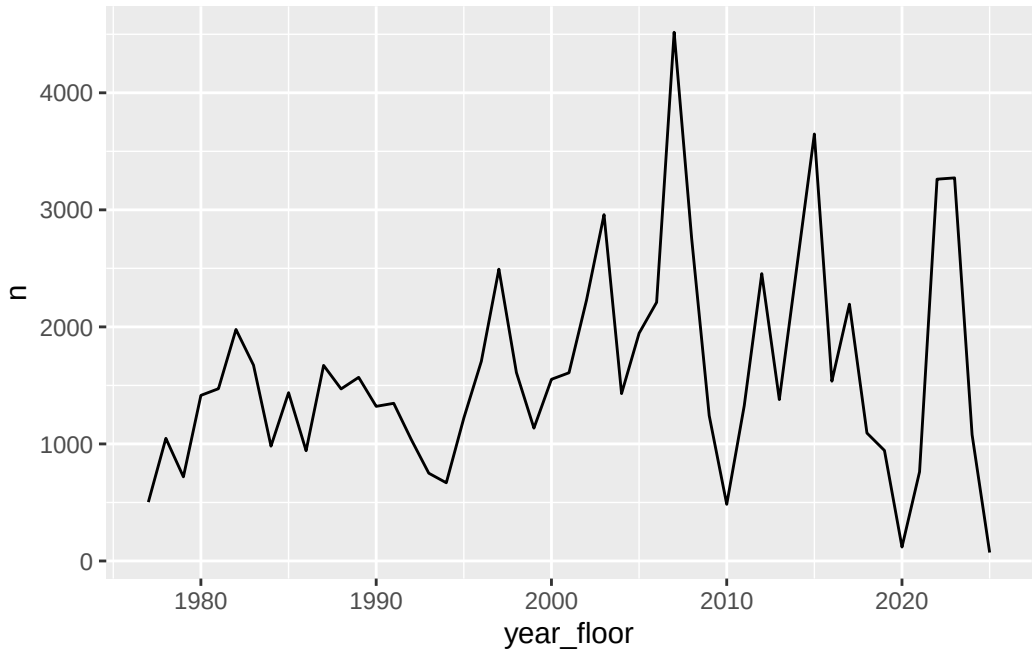
Use the `floor_date` function to round each date down to the first day of the month, which is great for time series grouping.

```
portal_rodent %>%
  mutate(month_floor = floor_date(date, "month")) %>%
  count(month_floor) %>%
  ggplot(aes(x = month_floor, y = n)) +
  geom_line()
```



Alternatively, use the `floor_date` function to round each date down to first day of the year:

```
portal_rodent %>%
  mutate(year_floor = floor_date(date, "year")) %>%
  count(year_floor) %>%
  ggplot(aes(x = year_floor, y = n)) +
  geom_line()
```


4000
3000
2000
1000
0
n
1980
1990
2000
2010
2020
year_floor