Working with Dates

Author: Cole Brookson **Date:** 20 July 2022

Along with factors, dates are one of the other datatypes that can be a nuisance to work with. EEB-type samples are often taken at inconsistent sampling intervals, and we don't always keep this information in concise formats.

We'll focus here on a few key activities to do with dates:

1. Taking a date data type and extracting sub-components from it (i.e. year, month, day, week)
2. Turning a non-date data type into a date

Extracting Date Components

Let's use an example of a timeseries with data collected through time on precipitation and temperature in Alaska:

```
library(tidyverse)
library(lterdatasampler)

df <- lterdatasampler::arc_weather
```

We can see what we're dealiang with here:

```
head(df)
```

```
## # A tibble: 6 x 5
##   date       station              mean_airtemp daily_precip mean_windspeed
##   <date>     <chr>                       <dbl>        <dbl>          <dbl>
## 1 1988-06-01 Toolik Field Station          8.4          0               NA
## 2 1988-06-02 Toolik Field Station          6            0               NA
## 3 1988-06-03 Toolik Field Station          5.8          0               NA
## 4 1988-06-04 Toolik Field Station          1.8          0               NA
## 5 1988-06-05 Toolik Field Station          6.8          2.5             NA
## 6 1988-06-06 Toolik Field Station          5.2          0               NA
```

So we already have a date column of the special data type `date`. This in fact makes our life easy, as it's always easier to extract information from a pre-formatted date column.

The best tool in our toolbox for this type of task is the `lubridate` package. This package has a ton of great functions that let us work with dates more easily. Let's test it out. Say we want to make a vector that has just all the years extracted from our date column. We could do that very easily like this with the `lubridate::year()` function:

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
year <- lubridate::year(df$date)

# print the first ten entries
year[1:10]
```

```
##  [1] 1988 1988 1988 1988 1988 1988 1988 1988 1988 1988
```

And we see that this worked! So the way we probably use this the most often is to make new columns in a dataframe. Let's go ahead and make a new column each for year, month, week, and day in our `df` dataframe:

```
df %>%
  # since we want lubridate to work with each row individually, use rowwise()
  dplyr::rowwise() %>%
  # we use mutate() to make a new column
  dplyr::mutate(
    year = lubridate::year(date),
    month = lubridate::month(date),
    week = lubridate::week(date),
    day = lubridate::day(date)
  )
```

```
## # A tibble: 11,171 x 9
## # Rowwise:
##    date       station mean_airtemp daily_precip mean_windspeed  year month  week
##    <date>     <chr>          <dbl>        <dbl>          <dbl> <dbl> <dbl> <dbl>
##  1 1988-06-01 Toolik~          8.4          0               NA  1988     6    22
##  2 1988-06-02 Toolik~          6            0               NA  1988     6    22
##  3 1988-06-03 Toolik~          5.8          0               NA  1988     6    23
##  4 1988-06-04 Toolik~          1.8          0               NA  1988     6    23
##  5 1988-06-05 Toolik~          6.8          2.5             NA  1988     6    23
##  6 1988-06-06 Toolik~          5.2          0               NA  1988     6    23
##  7 1988-06-07 Toolik~          2.2          7.6             NA  1988     6    23
##  8 1988-06-08 Toolik~          9.4          0               NA  1988     6    23
##  9 1988-06-09 Toolik~         13.1          0               NA  1988     6    23
## 10 1988-06-10 Toolik~         17.7          0                3.9  1988     6    24
## # ... with 11,161 more rows, and 1 more variable: day <int>
```

And here we can see the new columns have been made for us!

Forming Dates

To start thinking about forming dates, we'll use some fake data to make our lives easier. We can imagine the opposite scenario to above, we have some entries for, let's say, year and month, but no full date.

This brings about a somewhat more challenging problem as there's a decision point that needs to be executed – what day should we default to? This is a question that deserves careful consideration for each problem that arises and there is no one-size-fits-all solution. However, assuming you have decided that there is a simple assumption you can make (e.g. you will assume the data were collected on the first of the month), we can use this to make a new `date` column from our existing data.

Let's generate some fake data to work with:

```
df = data.frame(
  year = sample(c(2010:2020), replace = TRUE, 200),
  month = sample(c(1:12), replace = TRUE, 200),
  # we'll make a set of fake sampled data here
  observation = sample(c(12.5:16.6), replace = TRUE, 200)
)
```

The first thing to do is inpute our decided day values of the first day of each month. That's easy enough:

```
df = df %>%
  dplyr::mutate(
    day = 1
  )
df
```

```
##      year month observation day
## 1    2017     8        12.5   1
## 2    2011     4        12.5   1
## 3    2019    12        13.5   1
## 4    2019     2        13.5   1
## 5    2013     5        15.5   1
## 6    2014     9        15.5   1
## 7    2010     5        14.5   1
## 8    2020     3        12.5   1
## 9    2016     1        15.5   1
## 10   2013     5        15.5   1
## 11   2019    12        12.5   1
## 12   2018     5        13.5   1
## 13   2011    11        15.5   1
## 14   2014     2        16.5   1
## 15   2011     6        16.5   1
## 16   2012     4        14.5   1
## 17   2010     3        12.5   1
## 18   2018     7        14.5   1
## 19   2010     4        16.5   1
## 20   2012     4        16.5   1
## 21   2017    10        14.5   1
## 22   2011     1        14.5   1
## 23   2017     9        14.5   1
## 24   2013     3        13.5   1
## 25   2012     2        12.5   1
## 26   2015     9        13.5   1
## 27   2015     8        14.5   1
## 28   2012     3        12.5   1
## 29   2013     4        15.5   1
## 30   2017     6        15.5   1
## 31   2013    10        12.5   1
## 32   2019     1        16.5   1
## 33   2011     6        13.5   1
## 34   2010     3        13.5   1
## 35   2018     1        12.5   1
## 36   2015     5        16.5   1
## 37   2010     7        16.5   1
## 38   2014     1        15.5   1
## 39   2020    11        16.5   1
## 40   2011    11        15.5   1
## 41   2018    12        14.5   1
## 42   2012    10        16.5   1
## 43   2012     5        16.5   1
## 44   2019     9        14.5   1
## 45   2016     2        16.5   1
## 46   2018     6        12.5   1
## 47   2012     1        13.5   1
## 48   2015     6        12.5   1
## 49   2014     9        16.5   1
## 50   2014     7        13.5   1
## 51  2010     4         15.5   1
## 52  2013     9         13.5   1
## 53  2017    12         14.5   1
## 54  2014     3         15.5   1
## 55  2014    12         13.5   1
## 56  2015     1         12.5   1
## 57  2017     2         12.5   1
## 58  2012     8         16.5   1
## 59  2017     4         15.5   1
## 60  2015     6         14.5   1
## 61  2013    10         12.5   1
## 62  2020    10         14.5   1
## 63  2013    10         16.5   1
## 64  2017     2         12.5   1
## 65  2019     4         15.5   1
## 66  2018    12         13.5   1
## 67  2011     8         13.5   1
## 68  2012     9         14.5   1
## 69  2017     3         13.5   1
## 70  2019     8         12.5   1
## 71  2017     7         14.5   1
## 72  2016     2         12.5   1
## 73  2011    12         12.5   1
## 74  2019     8         13.5   1
## 75  2013     4         14.5   1
## 76  2016     3         14.5   1
## 77  2011     9         13.5   1
## 78  2014     4         13.5   1
## 79  2010    12         14.5   1
## 80  2012    12         16.5   1
## 81  2017    12         14.5   1
## 82  2018     2         16.5   1
## 83  2016    10         14.5   1
## 84  2019     3         15.5   1
## 85  2019     1         15.5   1
## 86  2017     5         13.5   1
## 87  2010     4         13.5   1
## 88  2020     3         16.5   1
## 89  2013    10         14.5   1
## 90  2011     3         14.5   1
## 91  2012    10         12.5   1
## 92  2014    12         13.5   1
## 93  2010    10         14.5   1
## 94  2017     9         15.5   1
## 95  2011     7         14.5   1
## 96  2015     2         16.5   1
## 97  2020    12         13.5   1
## 98  2018    12         13.5   1
## 99  2016     1         13.5   1
## 100 2019    12         16.5   1
## 101 2010    10         13.5   1
## 102 2017     1         15.5   1
## 103 2015    11         13.5   1
## 104 2014     4         13.5   1
## 105 2020     9         15.5   1
## 106 2010     4         14.5   1
## 107 2014     1         15.5   1
## 108 2014     3         16.5   1
## 109 2019    12         15.5   1
## 110 2017     7         12.5   1
## 111 2015     2         16.5   1
## 112 2017    10         15.5   1
## 113 2012     8         13.5   1
## 114 2018     4         15.5   1
## 115 2017    11         13.5   1
## 116 2015     9         14.5   1
## 117 2014     3         15.5   1
## 118 2011     9         14.5   1
## 119 2017     4         15.5   1
## 120 2019     6         13.5   1
## 121 2010     1         15.5   1
## 122 2019    10         12.5   1
## 123 2018     2         14.5   1
## 124 2020    10         13.5   1
## 125 2015    11         12.5   1
## 126 2020     3         16.5   1
## 127 2015     6         16.5   1
## 128 2012     2         15.5   1
## 129 2018    11         15.5   1
## 130 2011    12         13.5   1
## 131 2016     4         15.5   1
## 132 2012     6         13.5   1
## 133 2014    11         12.5   1
## 134 2013     2         12.5   1
## 135 2013     5         13.5   1
## 136 2015     6         14.5   1
## 137 2013    12         15.5   1
## 138 2018     5         13.5   1
## 139 2019     7         14.5   1
## 140 2013    11         14.5   1
## 141 2020    12         12.5   1
## 142 2017    12         16.5   1
## 143 2018    10         12.5   1
## 144 2017     4         16.5   1
## 145 2010     1         16.5   1
## 146 2010     1         12.5   1
## 147 2012     9         16.5   1
## 148 2012    12         14.5   1
## 149 2015     5         16.5   1
## 150 2012     7         13.5   1
## 151 2011     1         14.5   1
## 152 2017     6         16.5   1
## 153 2015    11         13.5   1
## 154 2011     7         13.5   1
## 155 2011     4         14.5   1
## 156 2017     7         16.5   1
## 157 2010    12         14.5   1
## 158 2017    11         16.5   1
## 159 2011     5        13.5    1
## 160 2018     9        14.5    1
## 161 2019     3        14.5    1
## 162 2016     9        15.5    1
## 163 2013     4        15.5    1
## 164 2017     5        16.5    1
## 165 2018     2        13.5    1
## 166 2015     1        14.5    1
## 167 2013     5        16.5    1
## 168 2017    12        13.5    1
## 169 2018    12        12.5    1
## 170 2017    12        13.5    1
## 171 2020    12        15.5    1
## 172 2016     6        14.5    1
## 173 2015     2        13.5    1
## 174 2010     6        14.5    1
## 175 2019     7        13.5    1
## 176 2017     8        16.5    1
## 177 2018     9        13.5    1
## 178 2010     9        13.5    1
## 179 2018     2        13.5    1
## 180 2017     7        13.5    1
## 181 2020     9        16.5    1
## 182 2019     5        15.5    1
## 183 2020     7        12.5    1
## 184 2017     7        12.5    1
## 185 2011     4        12.5    1
## 186 2017     7        16.5    1
## 187 2019    12        12.5    1
## 188 2013     7        16.5    1
## 189 2020    12        15.5    1
## 190 2011    12        13.5    1
## 191 2013     6        13.5    1
## 192 2017    12        13.5    1
## 193 2016     3        16.5    1
## 194 2014    11        16.5    1
## 195 2020     3        14.5    1
## 196 2015     9        12.5    1
## 197 2013     8        12.5    1
## 198 2011     3        15.5    1
## 199 2019     8        16.5    1
## 200 2017     5        12.5    1
```

Great, we have the info we need. Now, we can go ahead and make a `date` column by combining our three other variables together using thye `lubrdiate::make_date()` function:

```
df = df %>%
  dplyr::rowwise() %>%
  dplyr::mutate(
    date = lubridate::make_date(year, month, day)
  )
df
```

```
## # A tibble: 200 x 5
## # Rowwise:
```

```
##     year month observation   day date
##    <int> <int>       <dbl> <dbl> <date>
##  1  2017     8        12.5     1 2017-08-01
##  2  2011     4        12.5     1 2011-04-01
##  3  2019    12        13.5     1 2019-12-01
##  4  2019     2        13.5     1 2019-02-01
##  5  2013     5        15.5     1 2013-05-01
##  6  2014     9        15.5     1 2014-09-01
##  7  2010     5        14.5     1 2010-05-01
##  8  2020     3        12.5     1 2020-03-01
##  9  2016     1        15.5     1 2016-01-01
## 10  2013     5        15.5     1 2013-05-01
## # ... with 190 more rows
```

Great, we can visually check this by looking across the first few rows and we see our function worked as it should.

Now you know how to move from dates to components and vice versa!