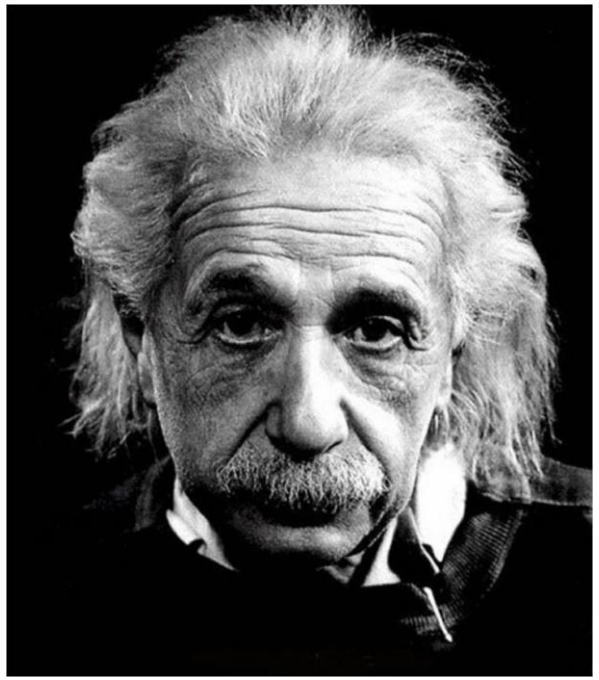
2021 Insurance Data Science Conference
The mondate package for R
Dan Murphy



Time is relative; its only worth depends upon what we do as it is passing.

— Albert Einstein

mondate Package
Agenda

- The ambiguity of the “month” unit of time per international standards
- R’s embracement of that ambiguity
- End-of-Business concept, not captured in the standards, is modeled by the mondate package based on principles debated in the area of Performance Management
- Simplifies aging of claims in units of months for forming triangles

What is a “month”?

International Organization for Standardization
Organisation internationale de normalization
Международная организация по стандартизации

ISO 8601: Date and time – Representations for information interchange

2004 (ISO 8601)

- **month** duration of **28, 29, 30, or 31** calendar days depending on the start and/or the end of the corresponding time interval within the specific calendar month
- **Note 1**: If the duration leads to a day that does not exist (like February 31), then “the ending calendar day has to be agreed on.”
- **Note 2**: In certain applications a month is considered as a duration of **30** calendar days

2019 (ISO 8601-01)

- ***month*** = *duration* (length) of a *calendar month*
- ***calendar month*** = a division of a *calendar year*
- ***calendar year*** = a *time scale unit* defined by a *calendar* system
- ***calendar*** = a time scale that uses *calendar day* as the basic unit

Broadly, beautifully, perfectly ambiguous

R's sequences by "month" can appear incongruous, but beautifully capture ISO 8601's ambiguity

- A month after 2021-03-01 is both 2021-03-29 and 2021-04-01

```
> seq(as.Date("2021-01-29"), length = 4, by = "months")
[1] "2021-01-29" "2021-03-01" "2021-03-29" "2021-04-29"
> seq(as.Date("2021-03-01"), length = 4, by = "months")
[1] "2021-03-01" "2021-04-01" "2021-05-01" "2021-06-01"
```

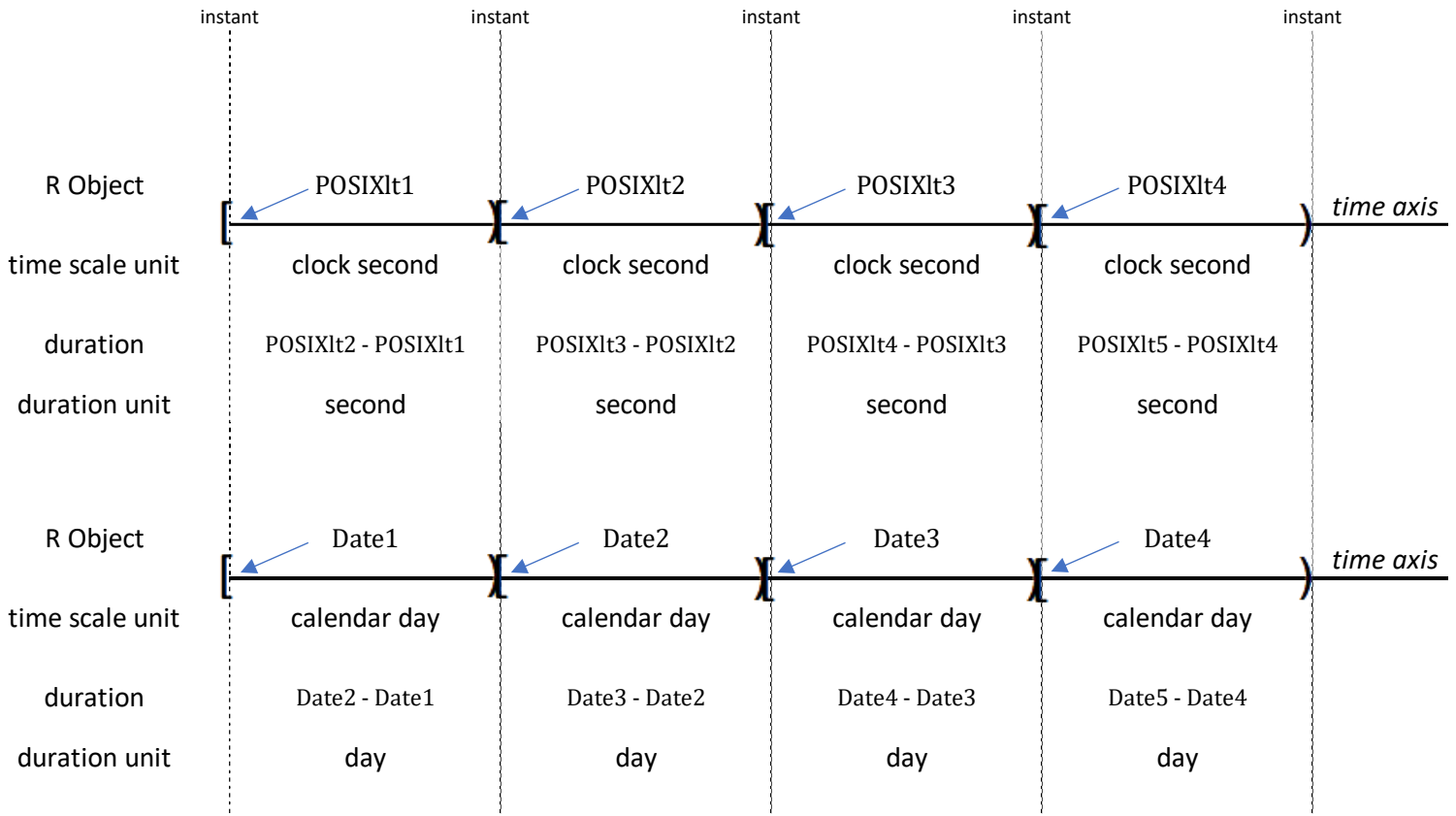
The standard is clear:
the duration of the **month** unit
– 28, 29, 30, or 31 days –
depends on where you start

- A year before 2020-03-01 is both 2019-03-01 and 2020-02-29

```
> seq(as.Date("2019-03-01"), length = 3, by = "years")
[1] "2019-03-01" "2020-03-01" "2021-03-01"
> seq(as.Date("2020-02-29"), length = 3, by = "years")
[1] "2020-02-29" "2021-03-01" "2022-03-01"
```

instant vs. span:
R's two major time objects, **POSIXt** and **Date**, point to the beginning of each day

- POSIXlt, POSIXct
- Date



- **Takeaway**: think of a "date" as both a span of time and an instant in time

Date's and POSIXt's beginning-of-day perspective does not model the concept **end of business**

- Accountants, and actuaries, have a well-honed understanding of what is meant by **end of business**
  - Year end
  - Month end
  - Month close
  - "Finish this project by the end of the day!" – could be an all-nighter ☹
- Example:
  - Last year end, YE 2020, was December 31, 2020
  - as.Date("2020-12-31") would capture YE2020 as the beginning of the day and therefore, without extra programming, would omit all activities after the dawn (time 00:00:00) of December 31, 2020
- Actuaries need an object to model **end of business**

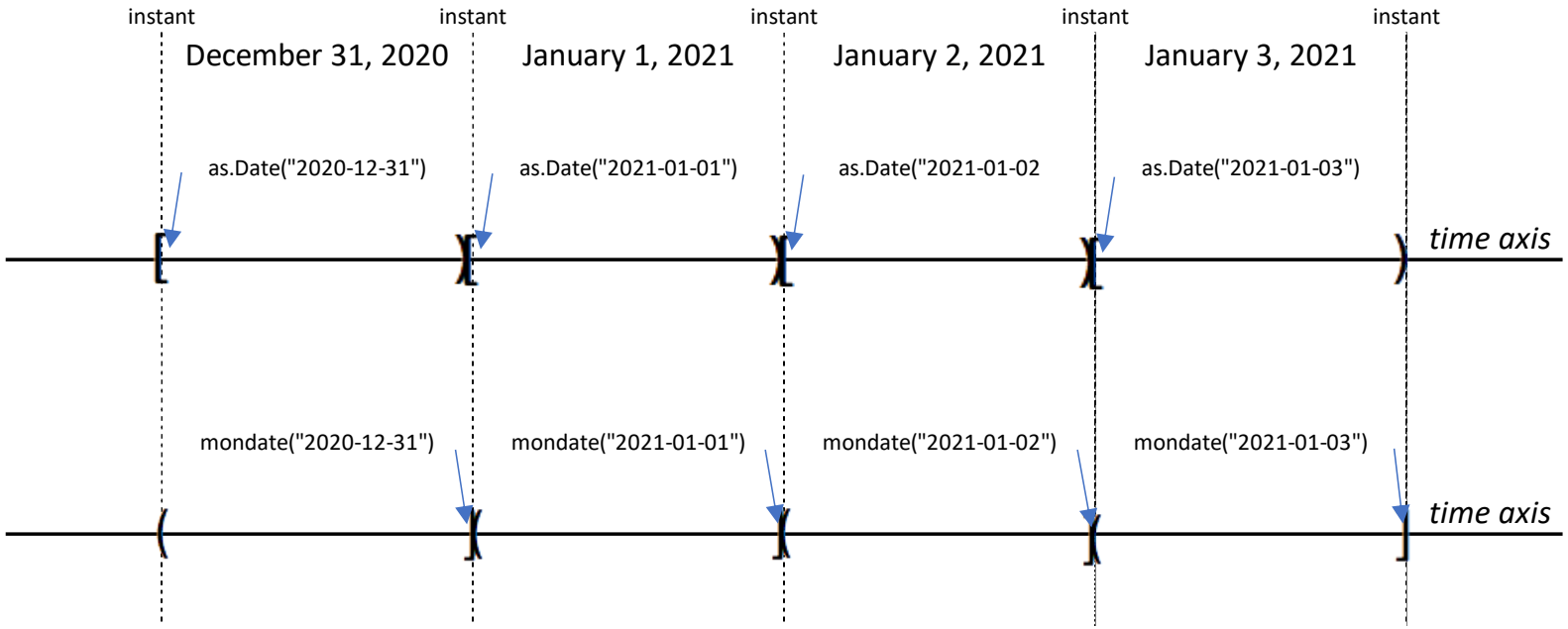
mondate (mŏn'•dāt) is an R time object that points to a day's **end of business**

- **Date**

  Time intervals are closed-left, open-right

- **mondate**

  Time intervals are open-left, closed-right*



- **Takeaway**: Pay attention to the instants in time associated with date/time objects when calculating age in units of "month"

* mimics R's `cut` function for numeric objects

mondate package on CRAN

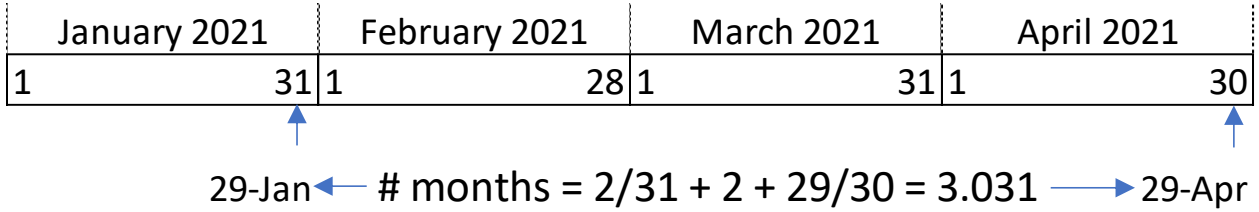
- `mondate(calendar day)` is a numeric value that holds the number of months from the end of 1999 to the end of business on that calendar day

```
> library(mondate)
> M = mondate("2000-12-31")
> print(M)
[1] 2000-12-31
> as.numeric(M)
[1] 12
> M + 12
mondate: timeunits="months"
[1] 2001-12-31
```

- Primary uses
  - Represent an as-of date or evaluation date
  - Calculate the age in months of something as of an evaluation date

mondate's fundamental formula

- **The number of months between the end of business on two dates**
  =
  the number of whole months between the dates
  +
  the fraction representing the proportion of the month remaining after the end of business on the first date
  +
  the fraction representing the proportion of the month passed as of the end of business on the second date




```
> mondate("2021-04-29") - mondate("2021-01-29")
Time difference of 3.031183 months
```

- Definition: 1 **year** = 12 months

~ Damien Laker, "Time Calculations for Annualizing Returns: The Need for Standardization", *The Journal of Performance Management*, 2008

Example 1
sequence of month-ends

- Using Date

```
> seq(as.Date("2021-01-31"), length = 4, by = "month")
[1] "2021-01-31" "2021-03-03" "2021-03-31" "2021-05-01"
```

- Using mondate

```
> seq(mondate("2021-01-31"), length = 4)
mondate: timeunits="months"
[1] 2021-01-31 2021-02-28 2021-03-31 2021-04-30
```

Example 2
age of Accident Year 2011 as of year end 2021

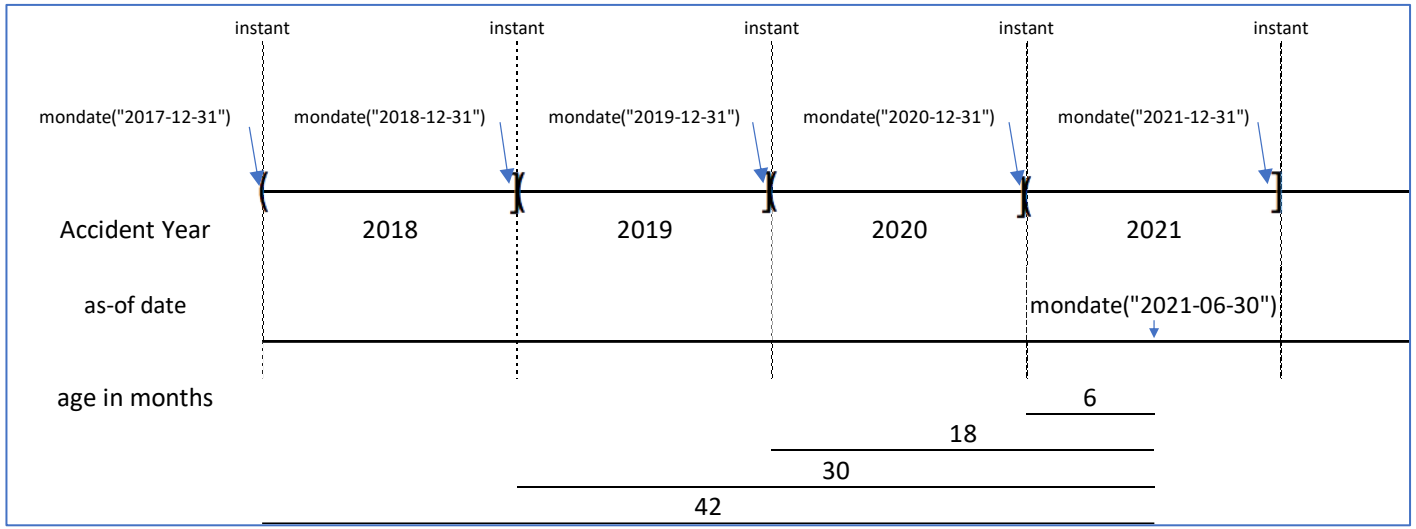
- Use a mondate to represent the as-of date December 31, 2021
- Use a mondate to represent the instant at the beginning of AY 2011
- The difference will be the age of the AY 2011 in months

```
> library(mondate)
> asof <- mondate("2021-12-31") # YE 2021
> AYbegin <- mondate("2010-12-31") # instant at beginning of AY 2011
> age <- asof - AYbegin
> age
Time difference of 132 months
```

Example 3
age of multiple accident years as of a mid-year evaluation date

- Use a mondate to represent the June 30th evaluation
- Introducing **mondate.ymd** "helper function" to represent the instant at the beginning of each accident year = instant at the end of the prior accident year
- Subtract

```
> AY = 2018:2021
> EvaluationDate = mondate("2021-06-30")
> age = EvaluationDate - mondate.ymd(AY-1)
> age
Time differences in months
[1] 42 30 18  6
```



`mondate.ymd(y, m, d)` = end of business on year y, month m, day d
`mondate.ymd(y, m)` = end of business on the last day of month m, year y
`mondate.ymd(y)` = end of business on the last day of year y

Example 4
form a triangle from a database of claims as of multiple evaluation dates

- Suppose you have a database of claims and you want to form an accident year triangle as of June 30 evaluation dates from 2017 to 2021

```
> head(database, 3)
    accidentdate   AY amount   evaldate
118   2017-01-11 2017   1248 2017-06-30
210   2017-05-06 2017   1107 2017-06-30
222   2017-01-03 2017   1501 2017-06-30
```

You have stacked your evaluations into one data.frame

- Calculate the age of the accident year of each claim as of the evaldates

```
> database$AYage = as.numeric(mondate(database$evaldate) - mondate.ymd(database$AY - 1))
```

- Use ChainLadder package to form the triangle of aggregated claim counts

```
> longtriangle <- aggregate(database["amount"], by = database[c("AY", "AYage")], FUN = sum)
> library(ChainLadder)
> as.triangle(database, origin = "AY", dev = "AYage", value = "amount")
      AYage
AY        6    18     30     42     54
  2017 7724 83096 120551 149006 171017
  2018 9278 73526 128150 172663     NA
  2019 6033 77767 125420     NA     NA
  2020 6501 74089     NA     NA     NA
  2021 6957    NA     NA     NA     NA
```

Other

- mondate can model “at ultimate” with infinite time

```
> mondate(Inf)
mondate: timeunits="months"
[1] Inf
```

- mondate displays dates in specified format (argument `displayFormat`), or local format if unspecified

```
> mondate("2021-06-16") # ISO format
mondate: timeunits="months"
[1] 2021-06-16
> mondate("6/16/2021")  # U.S. format
mondate: timeunits="months"
[1] 06/16/2021
> mondate("6/16/2021", displayFormat = "%Y-%m-%d") #
mondate: timeunits="months"
[1] 2021-06-16
> mondate.ymd(2018:2021) # run in U.S.
mondate: timeunits="months"
[1] 12/31/2018 12/31/2019 12/31/2020 12/31/2021
```

- local format is determined by `Sys.getlocale` function … needs work for non-U.S. formats!

```
> Sys.getlocale("LC_TIME")
[1] "English_United States.1252"
```

Please send chiefmurphy at gmail.com the value of Sys.getlocale(“LC_TIME”), your location, and your local date format. Thank you!

Summary

- ISO 8601 month durations are purposely ambiguous but difficult to work with when aging actuarial data as of evaluation dates in units of months

- The mondate package, based on Damien Laker's 2008 paper, simplifies the aging calculation for P&C/General Insurance actuaries

- On CRAN and hosted at https://github.com/chiefmurph/mondate

Dan Murphy
chiefmurphy at gmail