

Subsetting a window of observations

MANIPULATING TIME SERIES DATA IN R



Harrison Brown

Graduate Researcher in Geography

Time series windows

Windows:

- Subset of time series
- Inherits frequency from parent time series
- Defined by start and end point

Purpose of windows:

- View a specified range of data
- Focus in on years/events of interest
- Ignore observations at the "edge" of the data

Time series windows

```
card_prices
```

```
2013-01-01 4.88
2013-01-02 5.03
2013-01-03 5.11
...
2014-12-29 4.98
2014-12-30 5.01
2014-12-31 4.94
```

```
card_window <- stats::window(
  x = card_prices,
  start = "2013-05-01",
  end = "2013-07-01")
card_window
```

```
2013-05-01 4.98
2013-05-02 4.92
...
2013-07-30 5.04
2013-07-31 4.96
```

- `start` and `end` are inclusive

Subsetting by logical expressions

```
subset <-  
  index(card_price) >= "2013-05-01" &  
  index(card_price) <= "2013-07-31"
```

```
subset
```

```
[1] FALSE FALSE FALSE  
[4] FALSE FALSE FALSE  
...
```

```
card_subset <- card_price[subset]
```

```
card_subset
```

```
2013-05-01 4.98  
2013-05-02 4.92  
2013-05-03 5.21  
...  
2013-07-29 4.89  
2013-07-30 5.04  
2013-07-31 4.96
```

Extracting specific dates

```
card_price["2014-02-28"]
```

```
2014-02-28  
      5.02
```

```
class(index(card_price))
```

```
[1] "Date"
```

```
class("2014-02-28")
```

```
[1] "character"
```

Let's practice!

MANIPULATING TIME SERIES DATA IN R

Monthly and quarterly data

MANIPULATING TIME SERIES DATA IN R



Harrison Brown

Graduate Researcher in Geography

Dates and aggregated data

Dates:

- ISO 8601
- Largest -> smallest
- 2012-09-25

Aggregation:

- Monthly mean
- Weekly maximum
- Daily median
- Shows general trend and patterns in the data

Monthly data:

- Which date to use?
- Data for January, 2003:
 - 2003-01-01 ?
 - 2003-01-31 ?
 - 2003-01-15 ?
 - 2003-01

Year-month

Monthly data:

```
laborday_2022 <- as_date('2022-09-05')  
laborday_2021 <- as_date('2021-09-06')
```

```
lubridate::month(laborday_2022)
```

```
[1] 9
```

```
lubridate::month(laborday_2021)
```

```
[1] 9
```

`zoo::yearmon` class:

- year-month data

```
as.yearmon(laborday_2022)
```

```
[1] "Sep 2022"
```

```
as.yearmon(laborday_2021)
```

```
[1] "Sep 2021"
```

zoo::yearmon

```
zoo::as.yearmon()
```

```
as.yearmon('2007-10-31')
```

```
[1] "Oct 2007"
```

```
laborday_2021 <- as_date('2022-09-06')  
as.yearmon(laborday_2022)
```

```
[1] "Sep 2021"
```

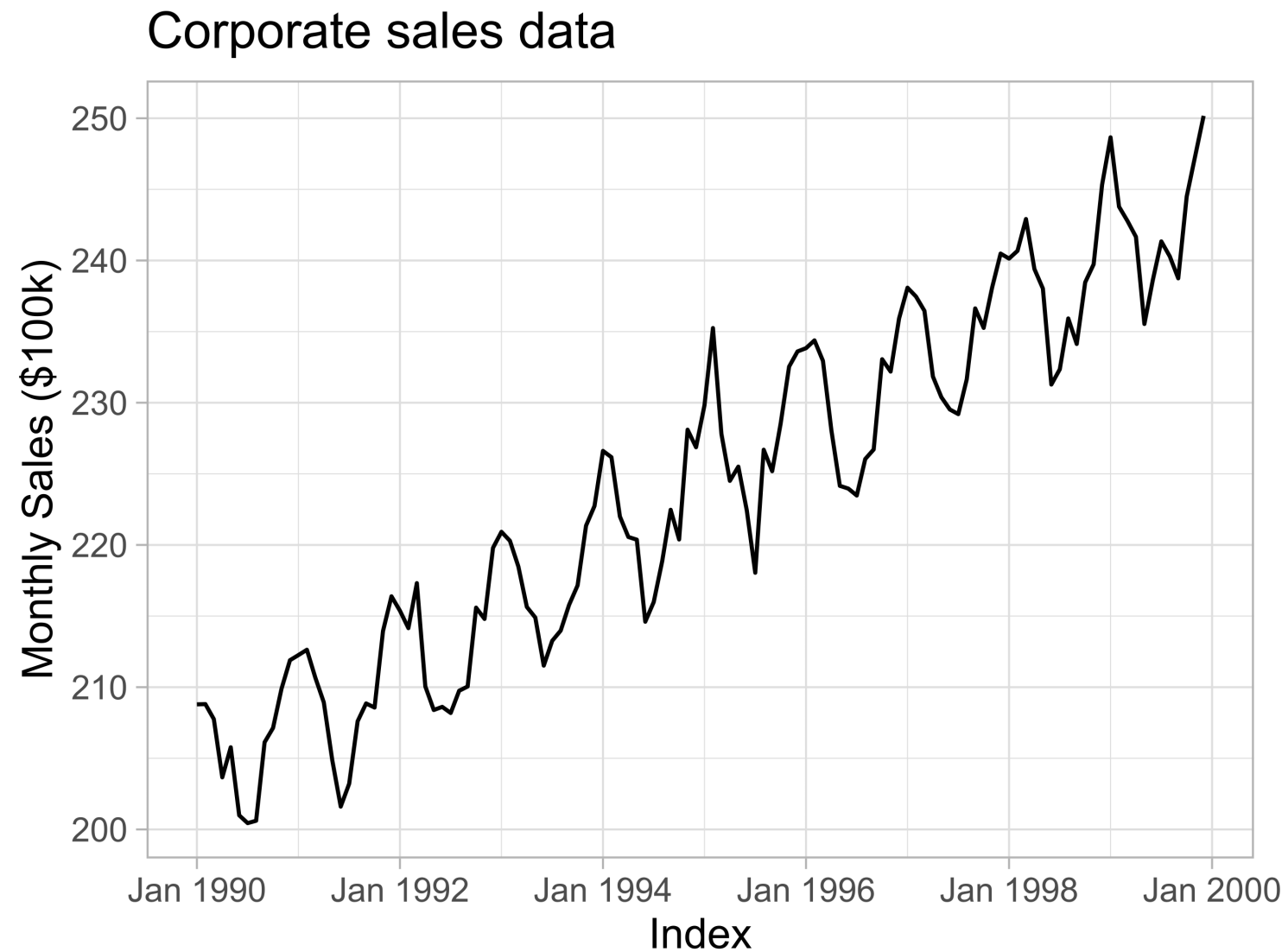
```
lunchtime <-  
  as.POSIXct('2022-09-21 13:30:00 EDT')  
as.yearmon(lunchtime)
```

```
[1] "Sep 2022"
```

```
as.yearmon(2008.245)
```

```
[1] "Mar 2008"
```

Monthly data



```
class(index(monthly_sales))
```

```
[1] "yearmon"
```

```
monthly_sales
```

```
Jan 1990 209.8921
Feb 1990 209.1493
Mar 1990 214.5236
Apr 1990 206.0842
May 1990 202.5655
...
```

zoo::yearqtr

```
zoo::yearqtr class
```

```
as.yearqtr(2018.639)
```

```
[1] "2018 Q3"
```

```
eid_2022 <- as_date('2022-05-01')  
as.yearqtr(eid_2022)
```

```
[1] "2022 Q2"
```

```
april_2003 <- as.yearmon("Apr 2003")  
as.yearqtr(april_2003)
```

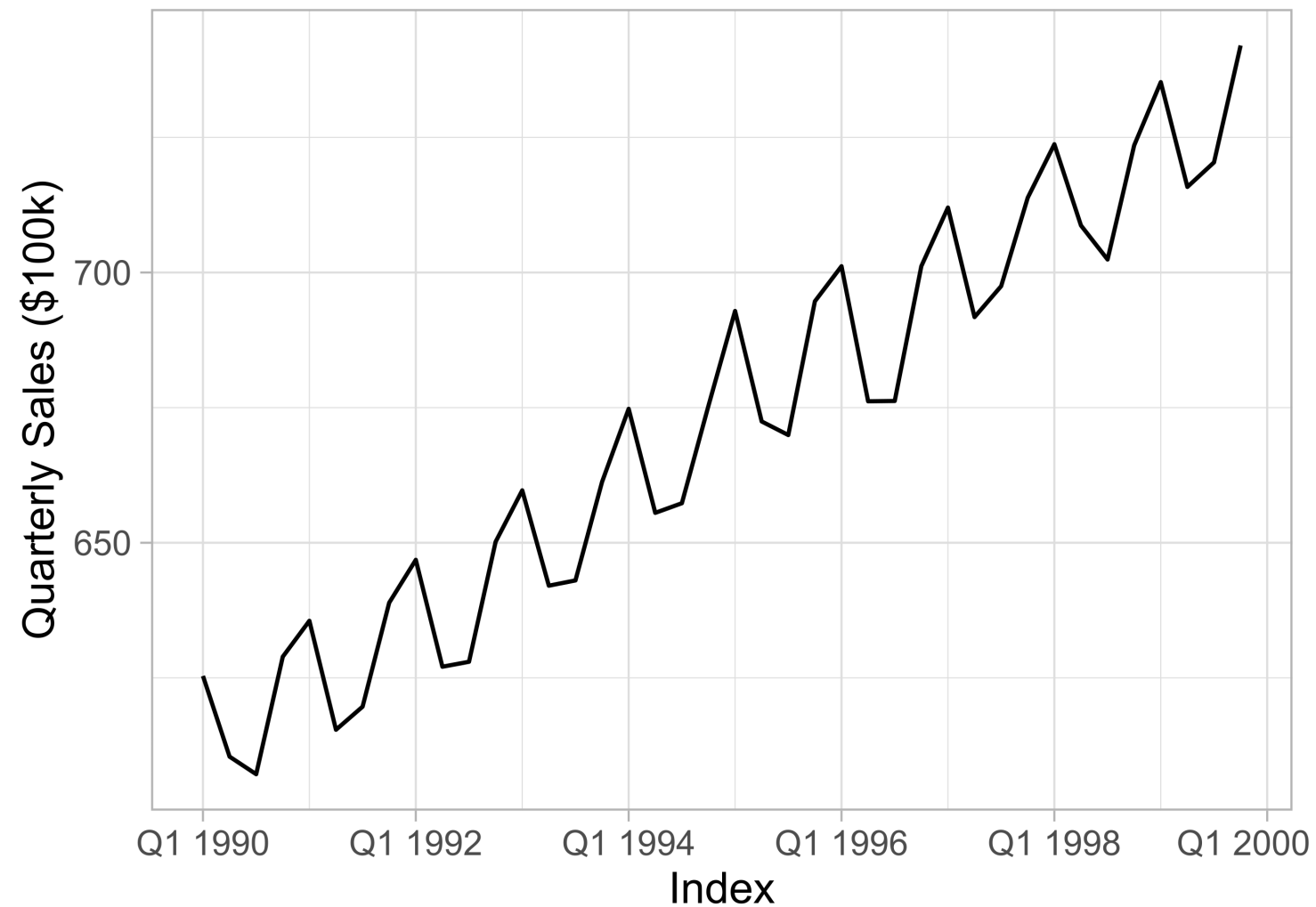
```
[1] "2003 Q2"
```

```
zoo::as.yearqtr('1990-10-30')
```

```
[1] NA
```

Quarterly data

Corporate sales data



```
quarterly_sales
```

```
1990 Q1 625.3594
1990 Q2 610.4269
1990 Q3 607.1768
1990 Q4 628.9146
1991 Q1 635.5531
...
```

```
class(index(quarterly_sales))
```

```
[1] "yearqtr"
```

Let's practice!

MANIPULATING TIME SERIES DATA IN R

Resampling and aggregating observations

MANIPULATING TIME SERIES DATA IN R



Harrison Brown

Graduate Researcher in Geography

Sampling frequency

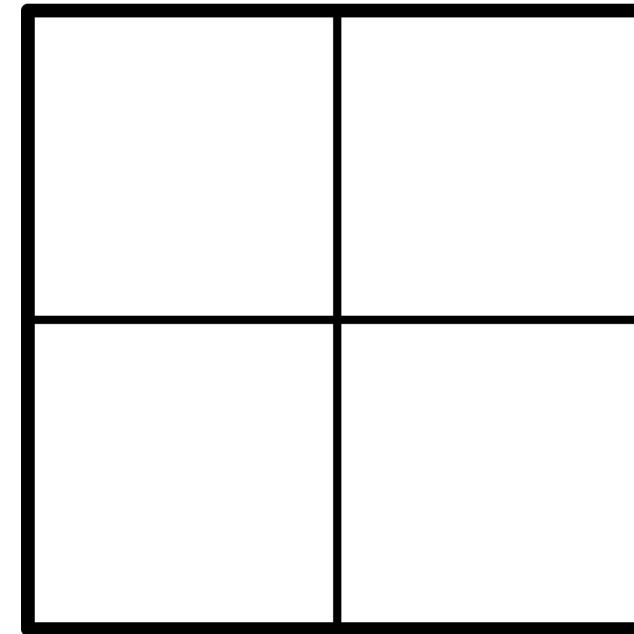
Frequency:

- Number of observations per year
- e.g., weekly, daily, monthly, ...

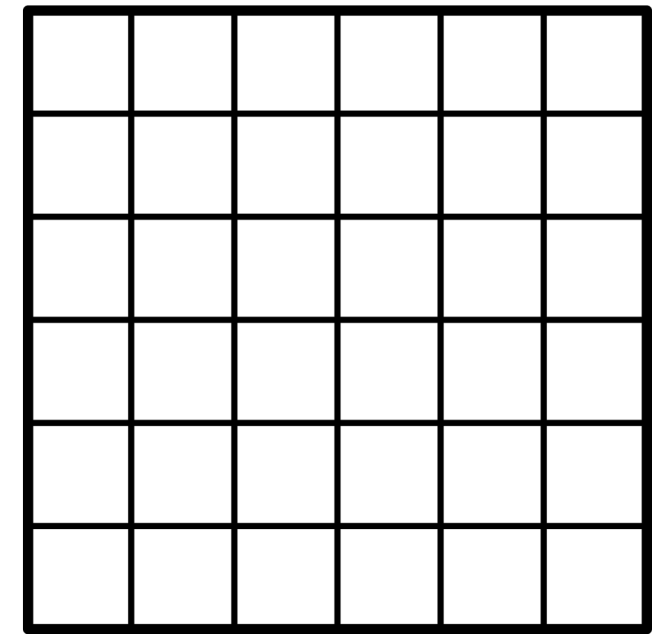
Temporal resolution:

- "High resolution" sampled often
- "Low resolution" sampled infrequently
- "High" and "low" are subjective

Low



High



Aggregation

- High resolution -> low resolution
- Applies a function like `mean` , `sum` , `max` to the chosen interval
- e.g.:
 - Monthly `sum` of daily data
 - Weekly `mean` of hourly values
 - ...
- Cannot 'reverse' aggregation
- Monthly total -> daily values?
- Provides statistics to describe patterns in the data
- Aggregation reduces information

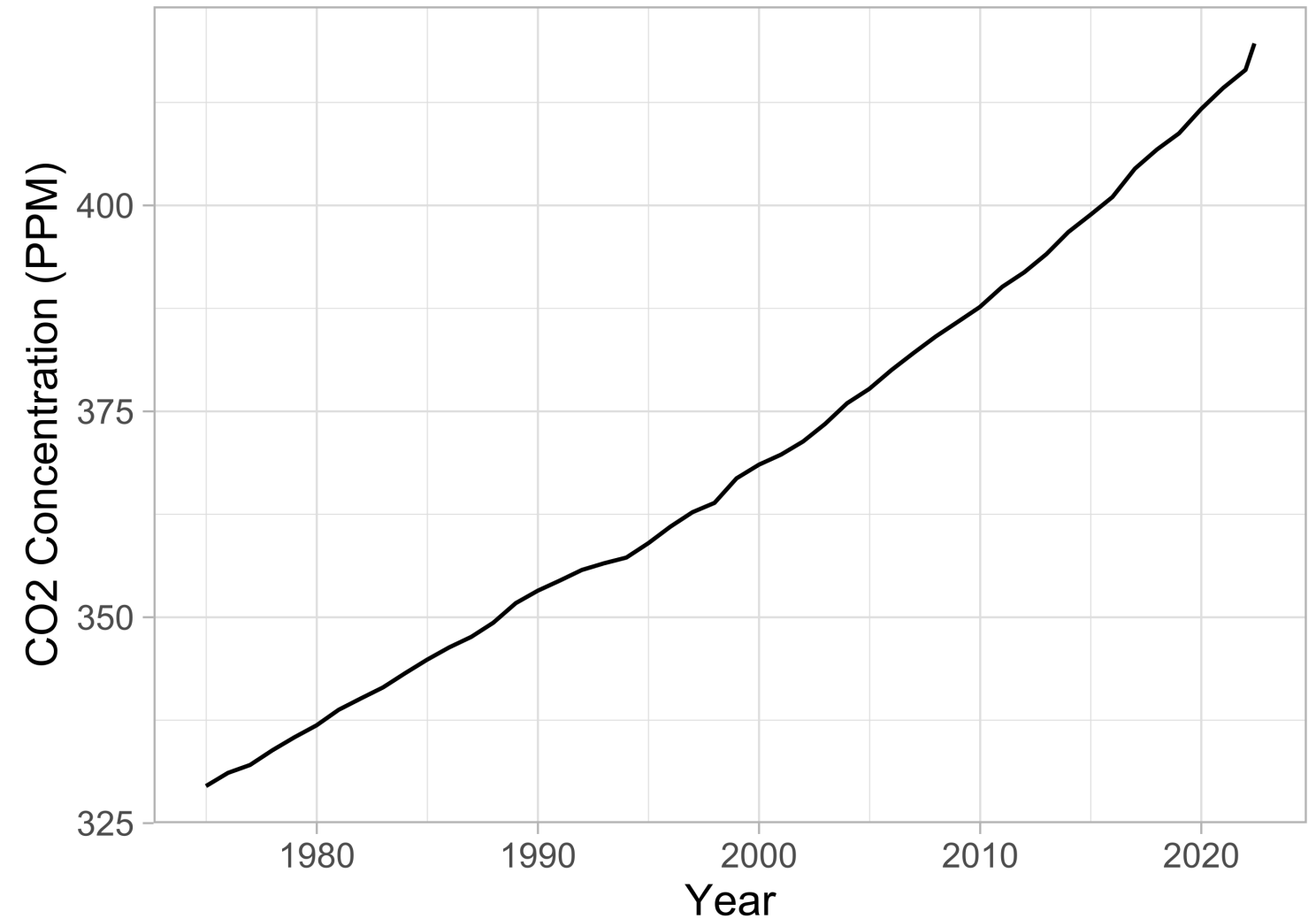
Aggregating data with xts

`xts` :

- eXtensible Time Series
- Extends the `zoo` package and `zoo` class of objects
- `apply.*()` functions

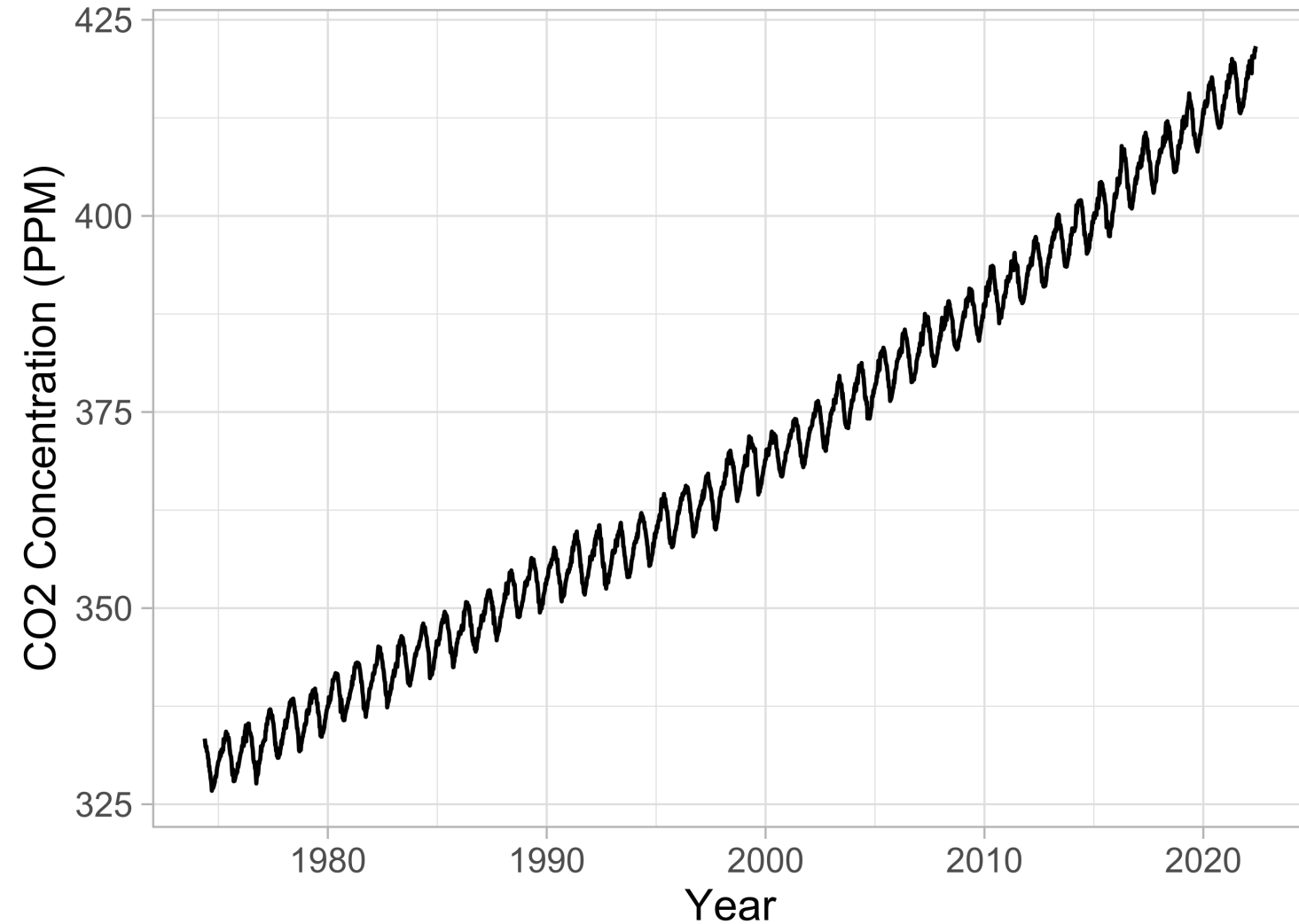
```
yearly_mean <-  
  apply.yearly(x = mauna Loa,  
               FUN = mean)  
autoplot(yearly_mean) +  
  labs(...)
```

Yearly Mean of CO2 Concentration

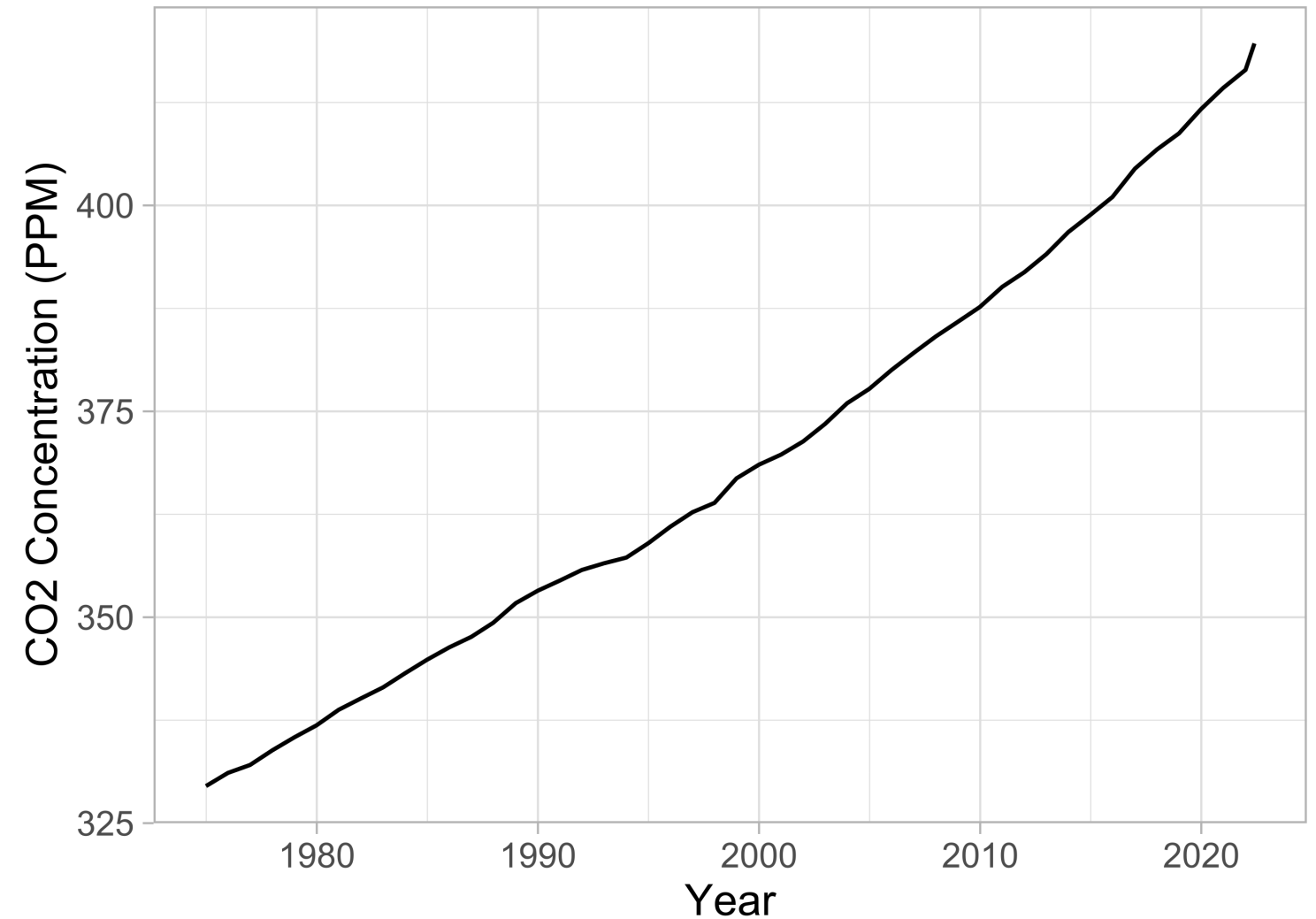


Aggregating data with xts

Mauna Loa CO2 Concentration



Yearly Mean of CO2 Concentration



apply-dot functions

```
daily_total <-  
  apply.daily(hourly_sales,  
              FUN = sum)
```

```
weekly_max <-  
  apply.weekly(daily_temperature,  
               FUN = max)
```

```
monthly_average <-  
  apply.monthly(daily_price,  
                FUN = mean)
```

```
apply.quarterly(sales_report,  
                FUN = sum)
```

```
apply.yearly(monthly_salary,  
              FUN = sum)
```

Endpoints and period.apply

```
xts::endpoints()
```

```
xts::period.apply()
```

```
biweekly_eps <-  
  endpoints(x = daily_data,  
            on = "weeks",  
            k = 2)
```

```
biweekly_data <-  
  period.apply(x = daily_data,  
              INDEX = biweekly_eps,  
              FUN = mean)
```

```
biweekly_data
```

```
2002-05-05 8.148611  
2002-05-19 8.146776  
2002-06-02 8.060020  
2002-06-16 8.028224  
2002-06-30 7.944792  
2002-07-14 7.930159  
...
```

Let's practice!

MANIPULATING TIME SERIES DATA IN R

Imputing missing values

MANIPULATING TIME SERIES DATA IN R



Harrison Brown

Graduate Researcher in Geography

Regular and irregular time series

Regular time series:

- No missing or `NA` values
- Even spacing between intervals

Real-world issues:

- Sensor and equipment failure
- Weather conditions
- ...

Aggregation:

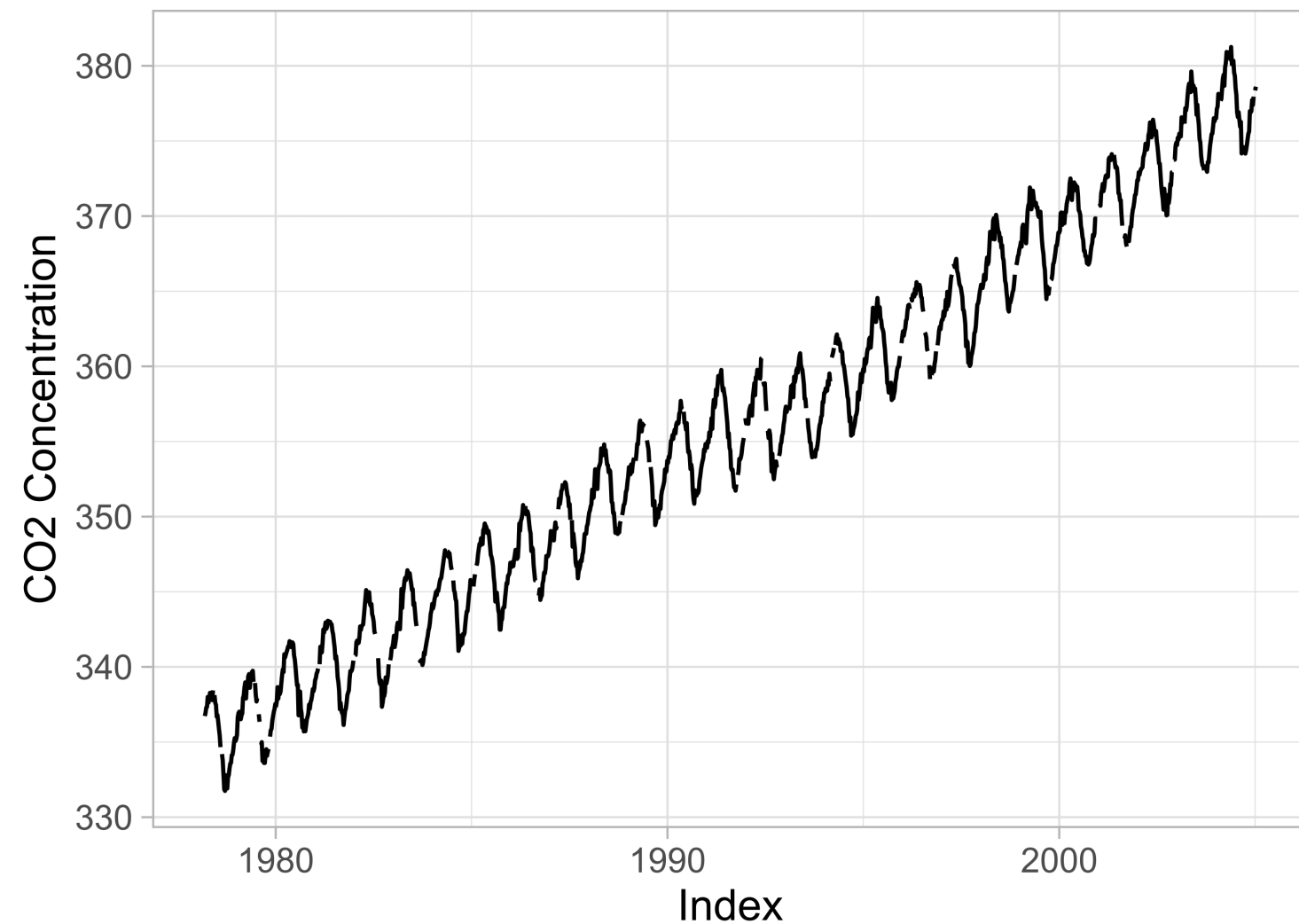
- Resamples data to lower temporal resolution
- Reduces information
- e.g. monthly sum of daily values

Imputation:

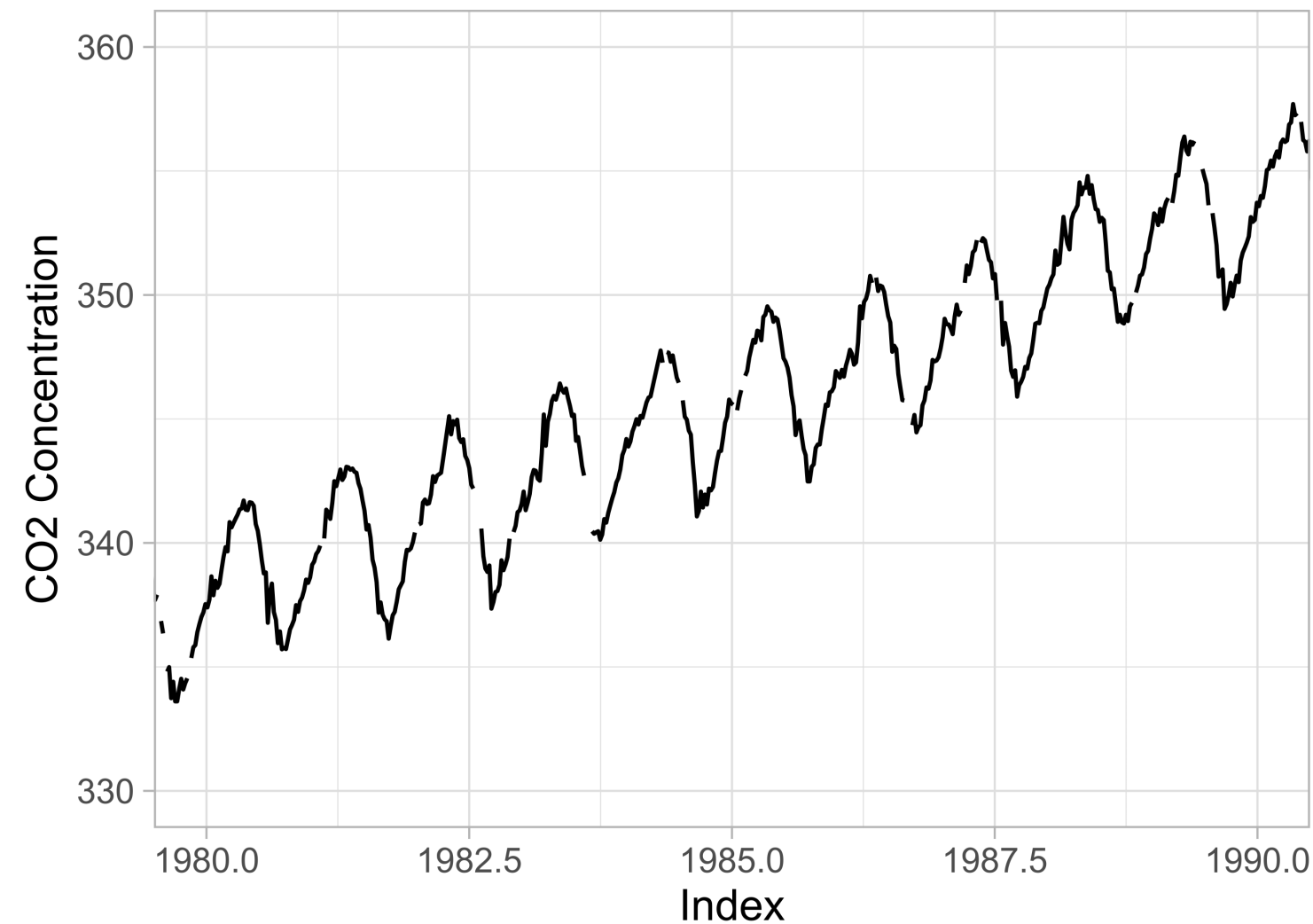
- Fills in missing values
- Different methods for determining values

Imputation

Mauna Loa, Missing Data Points



Mauna Loa, Missing Data Points



Imputing values with zoo

na-dot functions from `zoo` :

- `zoo::na.fill()`
- `zoo::na.locf()`
- `zoo::na.approx()`

Determining missing values

```
observations
```

```
2017-01-01 NA
2017-01-02  2
2017-01-03  2
2017-01-04  2
2017-01-05  4
2017-01-06  2
2017-01-07 NA
2017-01-08  1
2017-01-09  2
2017-01-10  2
...
```

```
sum(is.na(observations))
```

```
[1] 23
```

na.fill

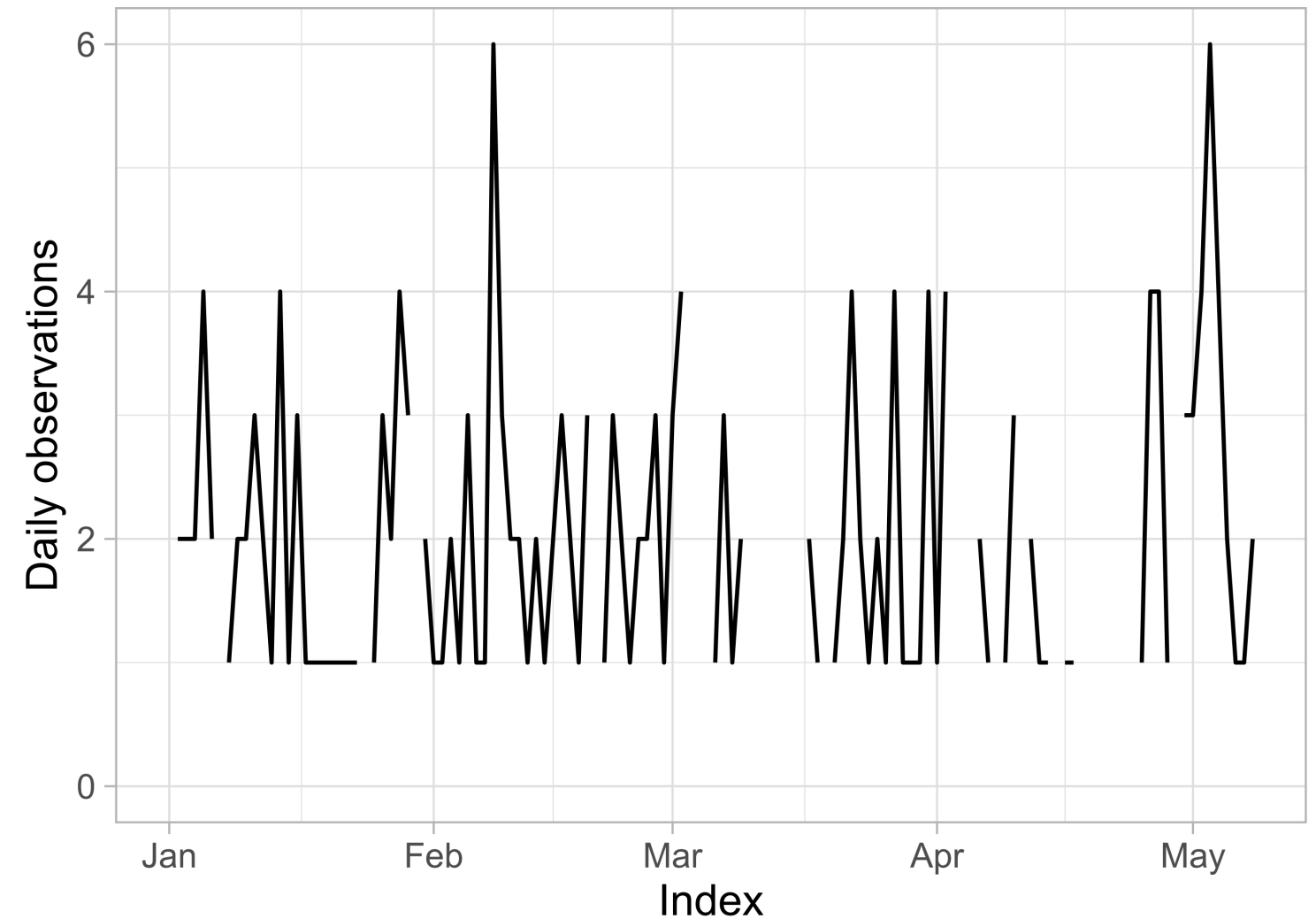
```
observations
```

```
2017-01-01 NA
2017-01-02  2
2017-01-03  2
2017-01-04  2
2017-01-05  4
...
```

```
table(observations, useNA = 'ifany')
```

1	2	3	4	5	6	<NA>
43	29	17	13	1	2	23

Observations, Missing Data Points



na.fill

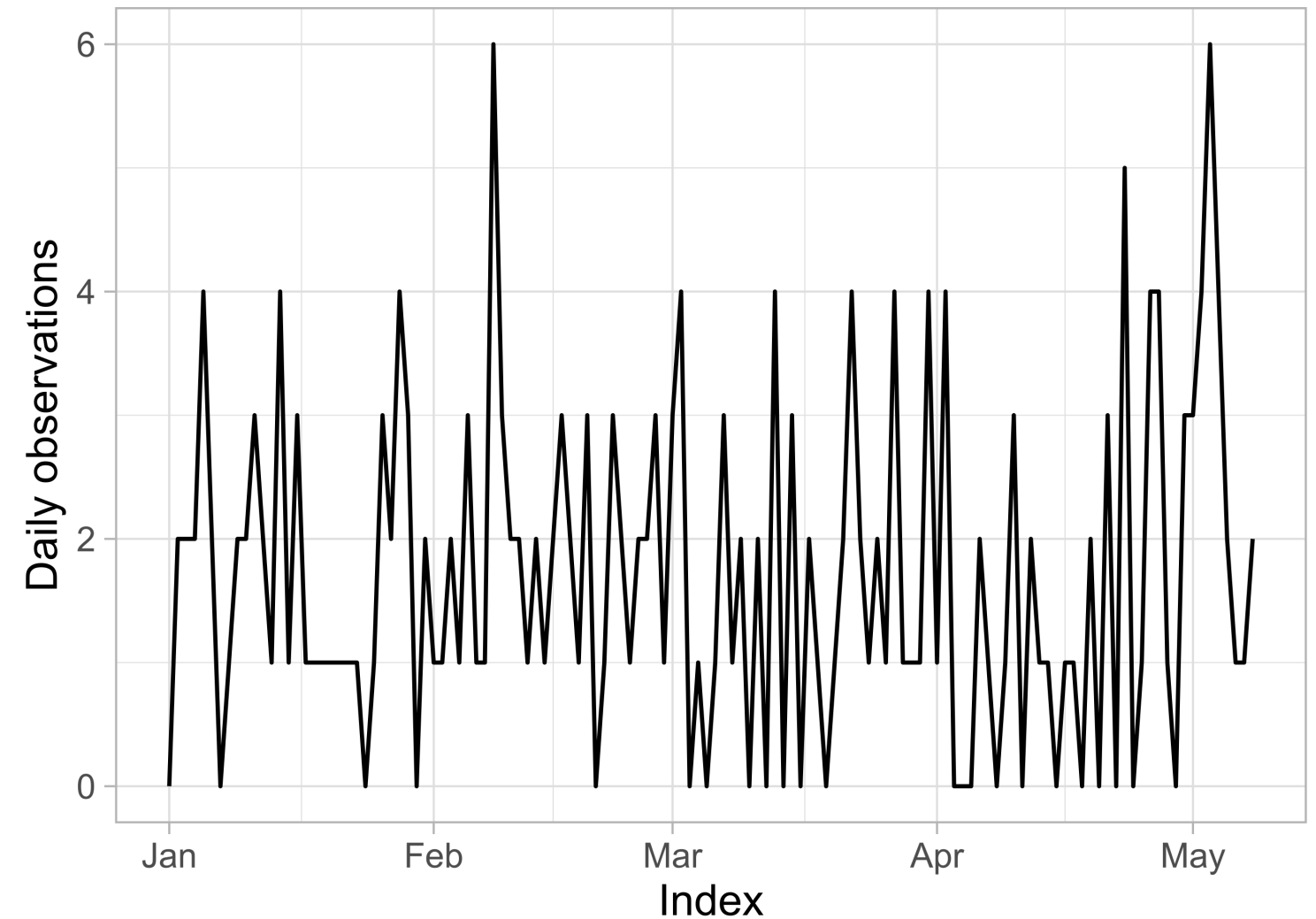
```
observations_fill <-  
  na.fill(object = observations,  
          fill = 0)
```

```
table(observations_fill)
```

```
 0   1   2   3   4   5   6  
23 43 29 17 13   1   2
```

```
autoplot(observations_fill)
```

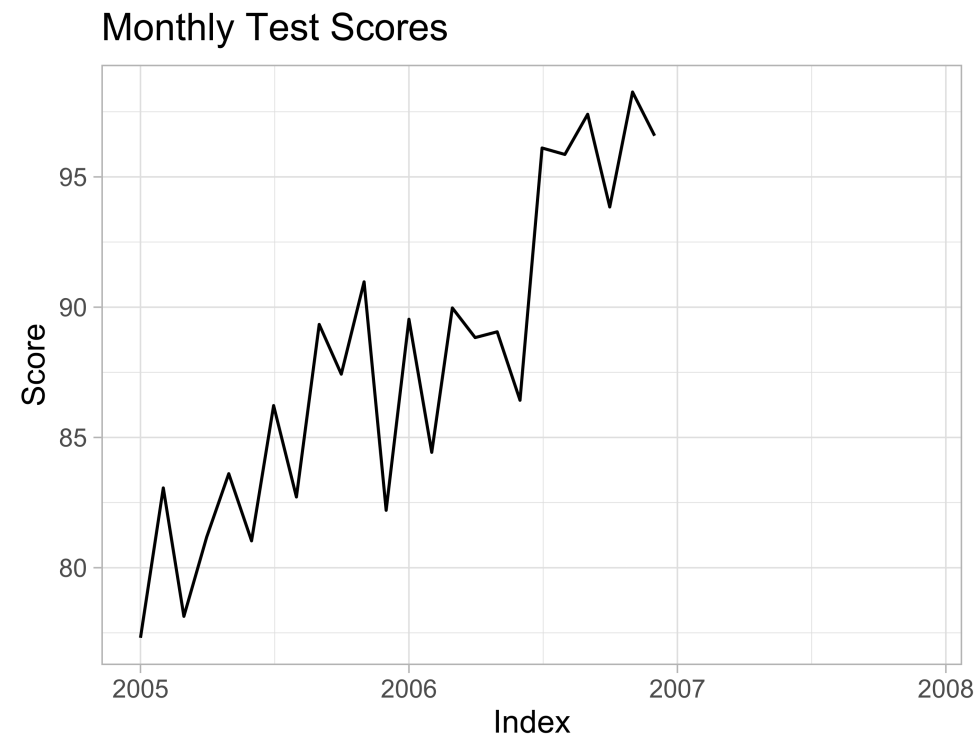
Daily Observations, Filled



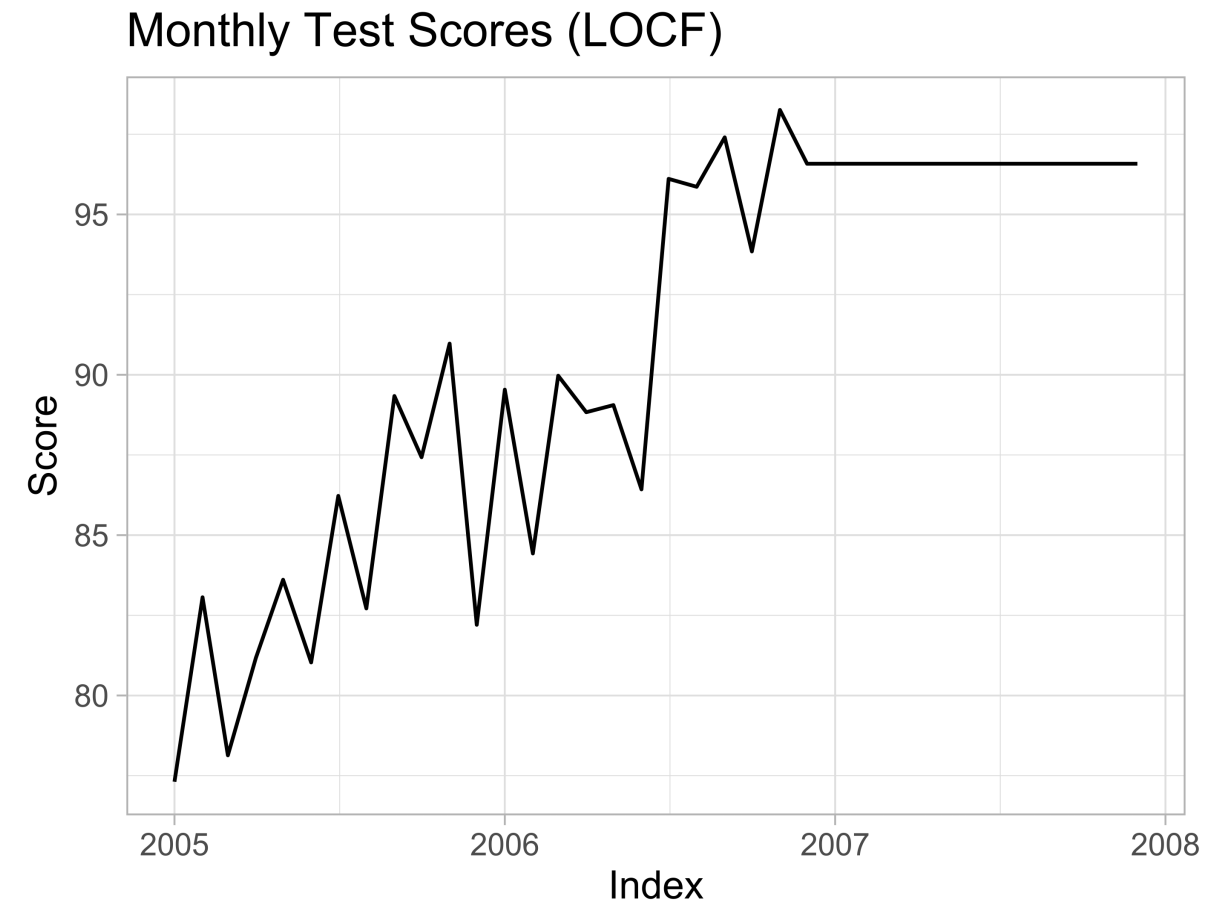
na.locf

```
autoplot(scores)
```

Warning message:
Removed 12 row(s) containing
missing values (geom_path).

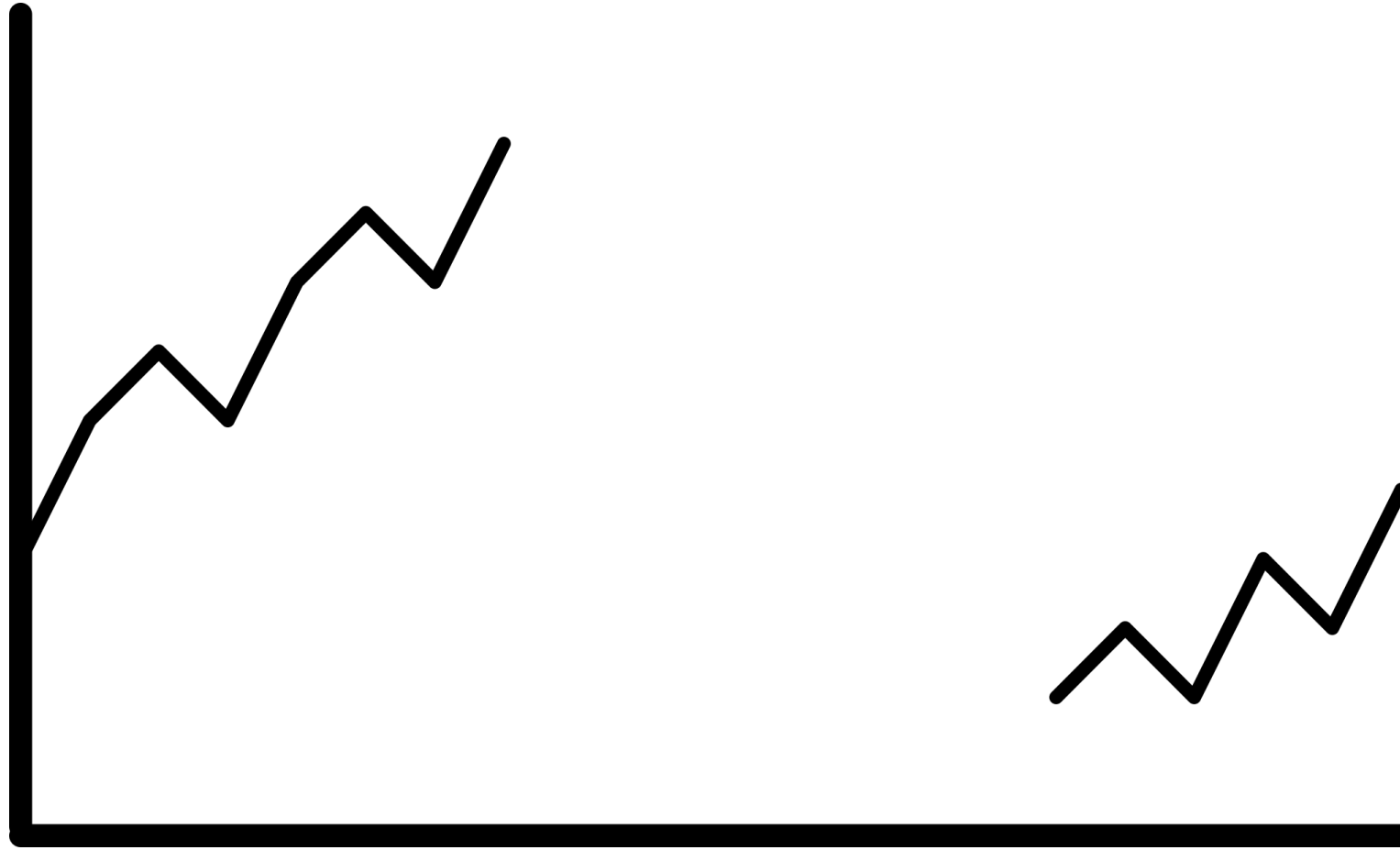


```
scores_locf <- na.locf(scores)  
autoplot(scores_locf)
```



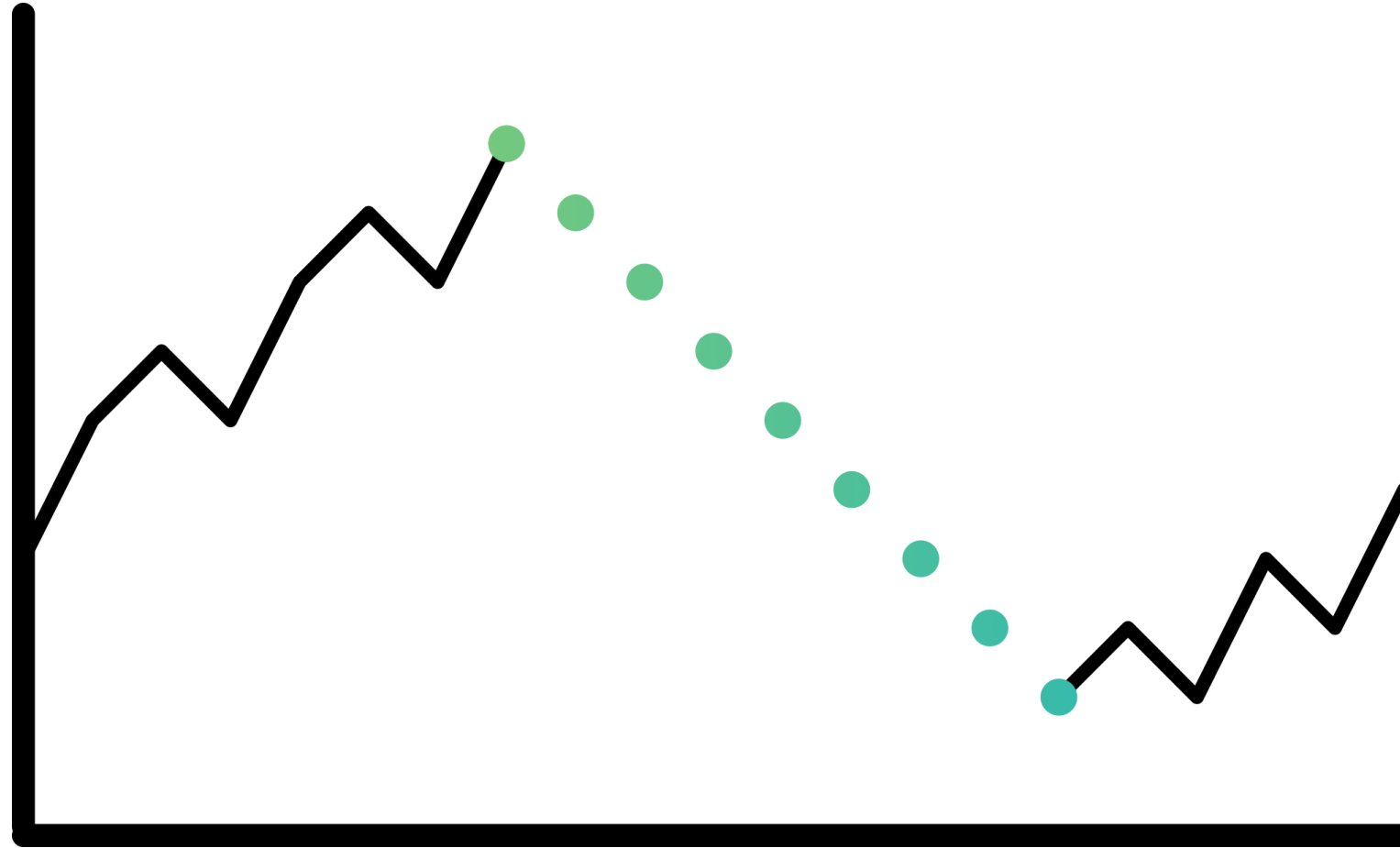
Linear interpolation

```
na.approx()
```



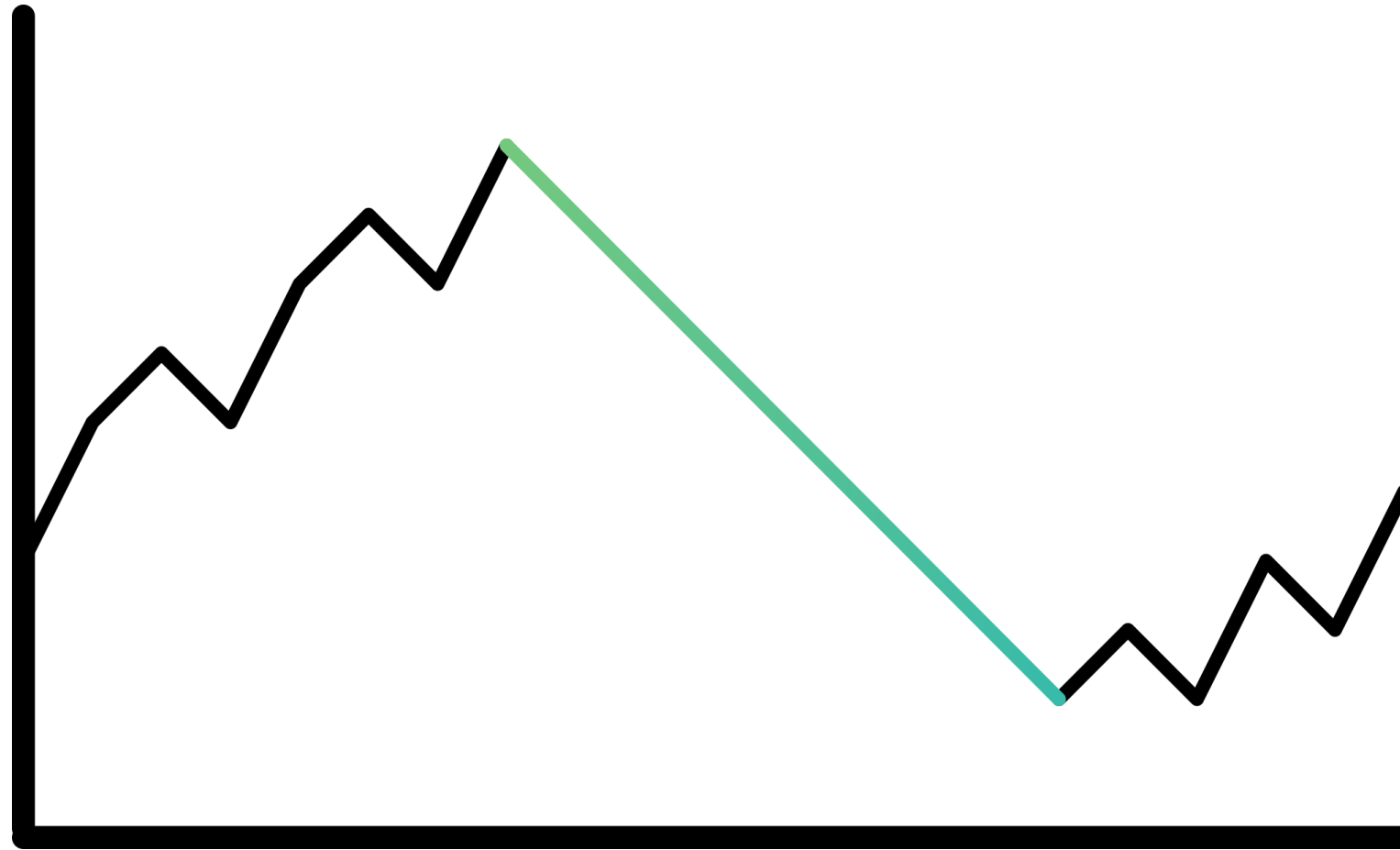
Linear interpolation

```
na.approx()
```



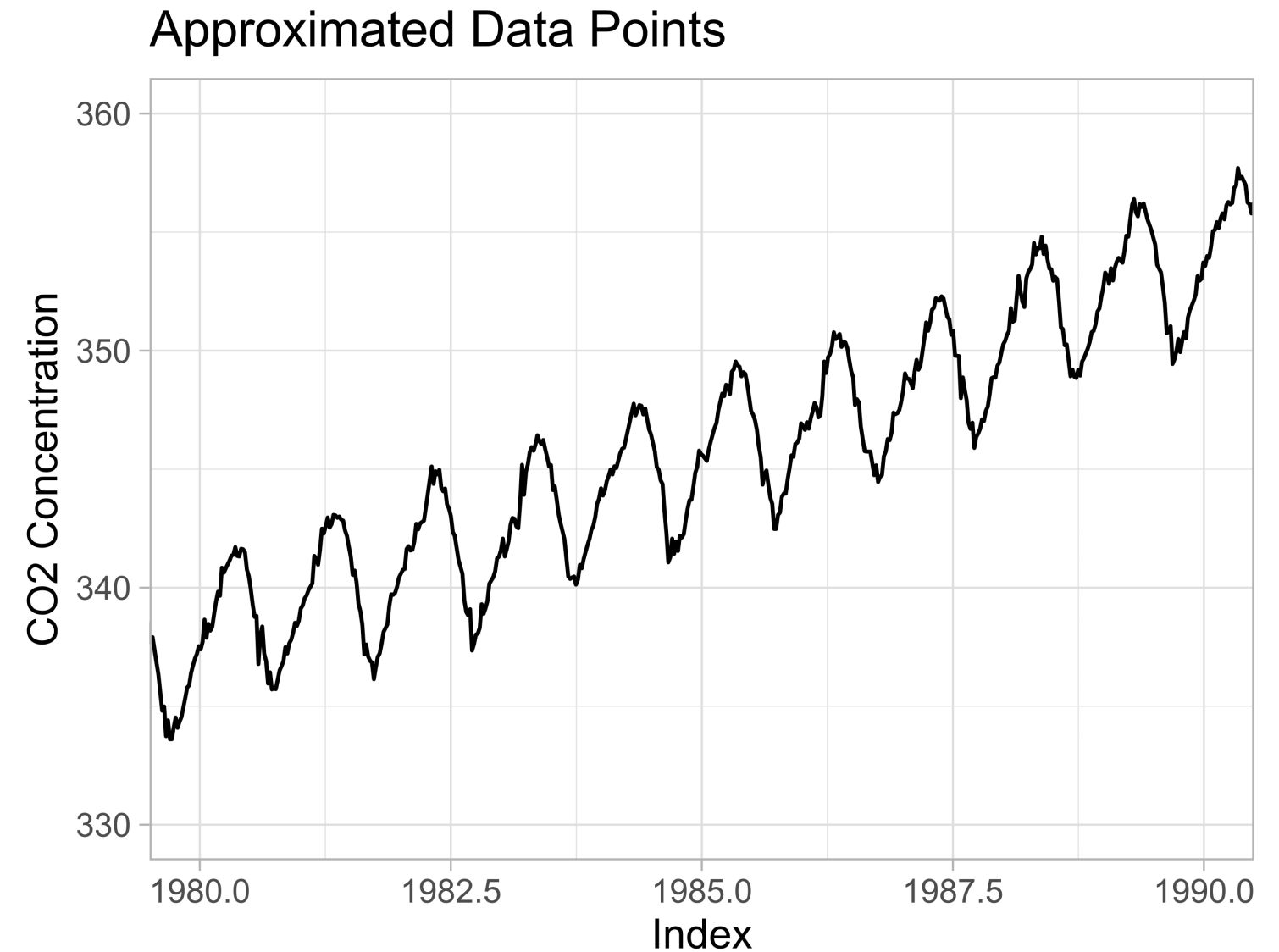
Linear interpolation

```
na.approx()
```



na.approx

```
maunaloa_approx <-  
  na.approx(maunaloa_missing)  
  
autoplot(maunaloa_approx) +  
  labs(  
    x = "Index",  
    y = "CO2 Concentration",  
    title = "Approximated Data Points"  
  )
```



Let's practice!

MANIPULATING TIME SERIES DATA IN R