SYS 5581 Project - Extract, Transform, and Load Data

Nick Coronato

Version of 2021-02-21 | Due 2021-02-22

Step 1: Identify a Time Series data set that you want to work with

For this project, I will be analyzing a set of exercise data for 186 patients.

Step 2: Acquire the data from its source location, reproducibly.

For this project, my data is stored on my local machine

Note: Ideally the data will be stored at and read from a Github repository. (Note: permission was granted to use this data, and no identifiable patient information is included in the raw data.)

```
url = 'https://github.com/uva-eng-time-series-sp21/coronato-nicholas/blob/main/CPET_raw_data.csv'

# To download data from Github, need to access the "Raw" version:
url <- "https://raw.githubusercontent.com/uva-eng-time-series-sp21/coronato-nicholas/main/CPET_raw_data2

# Alas, the token apparently expires after one download. So we'll load the file from local copy.

(CPET_raw  <-  read_delim("CPET_raw_data2.csv",",",
                          col_types = cols(.default = col_character(),
                                           "HR" = col_double(),
                                           "VO2" = col_double(),
                                           "VO2/kg" = col_double(),
                                           "VCO2" = col_double(),
                                           "RQ" = col_double(),
                                           "VE" = col_double(),
                                           "VE/VO2" = col_double(),
                                           "VE/VCO2" = col_double(),
                                           "Work" = col_double(),
                                           "PetO2" = col_double(),
                                           "PetCO2" = col_double(),
                                           "VEO22"= col_double(),
                                           "TMSPD" = col_double(),
                                           "TMELV" = col_double()
                          )))
```

```
## # A tibble: 16,564 x 30
##    PatientId SessionId Time  LocalTime TestLevel    HR SpO2    VO2 `VO2/kg`
##    <chr>     <chr>     <chr> <chr>     <chr>     <dbl> <chr> <dbl>    <dbl>
##  1 1         1         0:00~ 0:00:20   Baseline     74 <NA>  0.601      6.4
##  2 1         1         0:00~ 0:00:40   Baseline     74 <NA>  0.492      5.2
##  3 1         1         0:01~ 0:01:00   Baseline     73 <NA>  0.476      5
##  4 1         1         0:01~ 0:01:20   Baseline     74 <NA>  0.44       4.7
##  5 1         1         0:01~ 0:01:40   Baseline     75 <NA>  0.452      4.8
##  6 1         1         0:02~ 0:02:00   Baseline     74 <NA>  0.467      4.9
```

```
##  7 1         1         0:02~ 0:02:20   Baseline     78 <NA>  0.536      5.7
##  8 1         1         0:02~ 0:00:20   Exercise     86 <NA>  0.808      8.6
##  9 1         1         0:03~ 0:00:40   Exercise     86 <NA>  0.696      7.4
## 10 1         1         0:03~ 0:01:00   Exercise     86 <NA>  0.796      8.4
## # ... with 16,554 more rows, and 21 more variables: VCO2 <dbl>, RQ <dbl>,
## #   VE <dbl>, `VE/VO2` <dbl>, `VE/VCO2` <dbl>, Work <dbl>, PetO2 <dbl>,
## #   PetCO2 <dbl>, VEO22 <dbl>, FEO2 <chr>, FECO2 <chr>, RER <chr>, RR <chr>,
## #   METS <chr>, TMSPD <dbl>, TMELV <dbl>, Vtex <chr>, Vtin <chr>, Source <chr>,
## #   TypeUser <chr>, Summary <chr>
```

Step 3: Organize your data into a *tidy* data frame.

Organize by taking out the non-useful variables.

```
CPET_raw <- select(CPET_raw, -LocalTime, -FEO2, -FECO2, -RER,   -RR,    -METS,  -Vtex,  -Vtin, -Source,

CPET_raw %>% filter(PatientId != SessionId)
```

```
## # A tibble: 0 x 19
## # ... with 19 variables: PatientId <chr>, SessionId <chr>, Time <chr>,
## #   TestLevel <chr>, HR <dbl>, SpO2 <chr>, VO2 <dbl>, `VO2/kg` <dbl>,
## #   VCO2 <dbl>, RQ <dbl>, VE <dbl>, `VE/VO2` <dbl>, `VE/VCO2` <dbl>,
## #   Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>, TMSPD <dbl>,
## #   TMELV <dbl>
```

```
CPET_raw %>%
  select_if(function(x) !all(is.na(x)))
```

```
## # A tibble: 16,564 x 18
##    PatientId SessionId Time  TestLevel    HR   VO2 `VO2/kg`  VCO2    RQ    VE
##    <chr>     <chr>     <chr> <chr>     <dbl> <dbl>    <dbl> <dbl> <dbl> <dbl>
##  1 1         1         0:00~ Baseline     74 0.601      6.4 0.5    0.84  17.1
##  2 1         1         0:00~ Baseline     74 0.492      5.2 0.48   0.99  16.8
##  3 1         1         0:01~ Baseline     73 0.476      5   0.443  0.93  15.5
##  4 1         1         0:01~ Baseline     74 0.44       4.7 0.389  0.88  14.7
##  5 1         1         0:01~ Baseline     75 0.452      4.8 0.433  0.98  15.2
##  6 1         1         0:02~ Baseline     74 0.467      4.9 0.4    0.85  15.3
##  7 1         1         0:02~ Baseline     78 0.536      5.7 0.385  0.74  13.7
##  8 1         1         0:02~ Exercise     86 0.808      8.6 0.531  0.66  17.3
##  9 1         1         0:03~ Exercise     86 0.696      7.4 0.475  0.68  15.9
## 10 1         1         0:03~ Exercise     86 0.796      8.4 0.604  0.76  20.8
## # ... with 16,554 more rows, and 8 more variables: `VE/VO2` <dbl>,
## #   `VE/VCO2` <dbl>, Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>,
## #   TMSPD <dbl>, TMELV <dbl>
```

Make a new variable called Index so that each observation is individually identifiable, i.e. Session 1, Obs 1

```
#load package
require(data.table)

#  Turn data.frame into a data.table
CPET_ts2 <- data.table( CPET_raw )

#  Get running count by SessionId
CPET_ts2[ , Index := 1:.N , by = c("SessionId") ]
```

Make Index variable to be a two digit readout (i.e. 01, 02, . . .)

Convert time column into a more usable value (seconds instead of HH:MM:SS)

This can be used to create a dataframe of HR over time, per patient session.

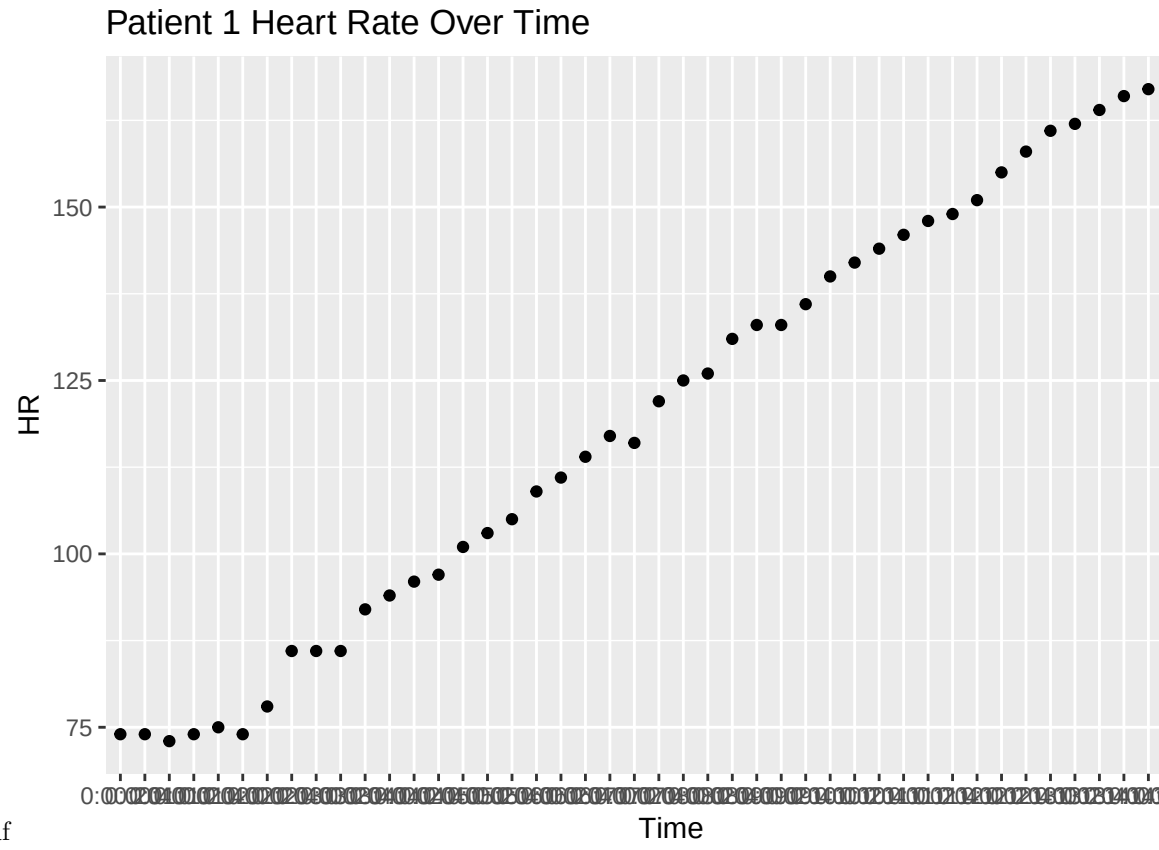
```
#This can be used to create a df of HR over time, per patient session

(CPET_ts %>%
  group_by(SessionId, NewTime) %>%
  summarise(HR) -> HR_by_patient)
```

```
## `summarise()` has grouped output by 'SessionId', 'NewTime'. You can override using the `.groups` argu
```

```
## # A tibble: 16,564 x 3
## # Groups:   SessionId, NewTime [16,562]
##    SessionId NewTime     HR
##    <chr>     <Period> <dbl>
##  1 1         20S         74
##  2 1         40S         74
##  3 1         1M 0S       73
##  4 1         1M 20S      74
##  5 1         1M 40S      75
##  6 1         2M 0S       74
##  7 1         2M 20S      78
##  8 1         2M 40S      86
##  9 1         3M 0S       86
## 10 1         3M 20S      86
## # ... with 16,554 more rows
```

This chunk is for example purposes; ggplot of Patient 1's heart rate over time.

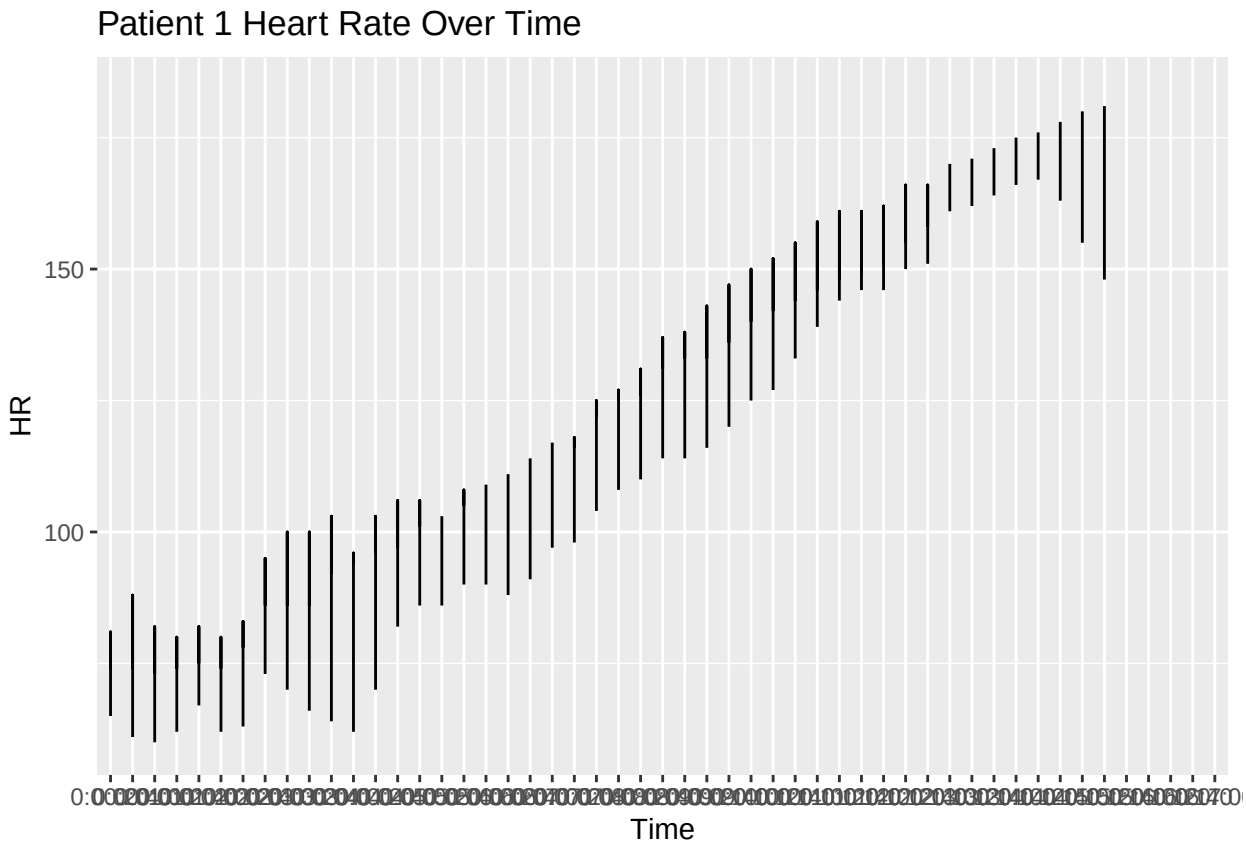


ggplot HR over time-1.pdf

Generate and print the tsibble.

```
## # A tsibble: 16,564 x 22 [1m 1s]
## # Key:       identifier [16,564]
##    PatientId SessionId identifier NewTime Time  TestLevel    HR SpO2    VO2
##    <chr>     <chr>     <chr>      <Perio> <chr> <chr>     <dbl> <chr> <dbl>
##  1 1         1         1.01       20S     0:00~ Baseline     74 <NA>  0.601
##  2 1         1         1.02       40S     0:00~ Baseline     74 <NA>  0.492
##  3 1         1         1.03       1M 0S   0:01~ Baseline     73 <NA>  0.476
##  4 1         1         1.04       1M 20S  0:01~ Baseline     74 <NA>  0.44
##  5 1         1         1.05       1M 40S  0:01~ Baseline     75 <NA>  0.452
##  6 1         1         1.06       2M 0S   0:02~ Baseline     74 <NA>  0.467
##  7 1         1         1.07       2M 20S  0:02~ Baseline     78 <NA>  0.536
##  8 1         1         1.08       2M 40S  0:02~ Exercise     86 <NA>  0.808
##  9 1         1         1.09       3M 0S   0:03~ Exercise     86 <NA>  0.696
## 10 1         1         1.10       3M 20S  0:03~ Exercise     86 <NA>  0.796
## # ... with 16,554 more rows, and 13 more variables: `VO2/kg` <dbl>, VCO2 <dbl>,
## #   RQ <dbl>, VE <dbl>, `VE/VO2` <dbl>, `VE/VCO2` <dbl>, Work <dbl>,
## #   PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>, TMSPD <dbl>, TMELV <dbl>,
## #   Index <chr>
```

```
CPET_tsbl %>%
  filter(PatientId %in% 1:3) %>%
  ggplot(mapping = aes(x = Time, y = HR)) +
  geom_line() +
  ggtitle("Patient 1 Heart Rate Over Time")
```



```r
interval(CPET_tsbl)
```

```
## <interval[1]>
## [1] 1m 1s
```

```r
is_regular(CPET_tsbl)
```

```
## [1] TRUE
```

```r
is_ordered(CPET_tsbl)
```

```
## [1] TRUE
```

```r
has_gaps(CPET_tsbl) %>% filter(.gaps == TRUE)
```

```
## # A tibble: 0 x 2
## # ... with 2 variables: identifier <chr>, .gaps <lgl>
```

Redo code

```r
# Confirm that values of PatiendId and SessionId are always identical:
CPET_raw %>% filter(PatientId != SessionId)  # Returns zero rows.
```

```
## # A tibble: 0 x 19
## # ... with 19 variables: PatientId <chr>, SessionId <chr>, Time <chr>,
## #   TestLevel <chr>, HR <dbl>, SpO2 <chr>, VO2 <dbl>, `VO2/kg` <dbl>,
## #   VCO2 <dbl>, RQ <dbl>, VE <dbl>, `VE/VO2` <dbl>, `VE/VCO2` <dbl>,
## #   Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>, TMSPD <dbl>,
## #   TMELV <dbl>
```

```r
CPET_raw %>%
  select(-SessionId) %>%                        # Drop redundant SessionId field
  select_if(function(x) !all(is.na(x))) %>%     # Drop fields containing only NA values
  mutate(TestLevel = as.factor(TestLevel)) %>%  # Convert data type from text to factors
  mutate(Time = lubridate::as.duration(hms(Time))) %>%  # Convert Time to duration type
  mutate(PatientId = as.integer(PatientId)) %>% # Convert PatientId to integer type
  select(PatientId, Time, everything()) -> CPET_rare

# Reference: https://r4ds.had.co.nz/dates-and-times.html#durations

CPET_rare %>% arrange(desc(PatientId), desc(Time))
```

```
## # A tibble: 16,564 x 17
##    PatientId Time                TestLevel    HR   VO2 `VO2/kg`  VCO2    RQ
##        <int> <Duration>          <fct>     <dbl> <dbl>    <dbl> <dbl> <dbl>
##  1       186 930s (~15.5 minutes) Recovery     65  0.68       NA  0.99    NA
##  2       186 900s (~15 minutes)   Recovery     73  0.84       NA  1.16    NA
##  3       186 870s (~14.5 minutes) Recovery     80  1.03       NA  1.39    NA
##  4       186 840s (~14 minutes)   Recovery     88  1.3        NA  1.55    NA
##  5       186 810s (~13.5 minutes) Exercise     90  1.38       NA  1.65    NA
##  6       186 780s (~13 minutes)   Exercise     89  1.37       NA  1.55    NA
##  7       186 750s (~12.5 minutes) Exercise     88  1.31       NA  1.41    NA
##  8       186 720s (~12 minutes)   Exercise     86  1.24       NA  1.3     NA
##  9       186 690s (~11.5 minutes) Exercise     84  1.19       NA  1.22    NA
## 10       186 660s (~11 minutes)   Exercise     82  1.13       NA  1.12    NA
```

```
## # ... with 16,554 more rows, and 9 more variables: VE <dbl>, `VE/VO2` <dbl>,
## #   `VE/VCO2` <dbl>, Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>,
## #   TMSPD <dbl>, TMELV <dbl>
```

```
CPET_rare %>%
  group_by(Time) %>%
  summarize(N = n())
```

```
## # A tibble: 1,603 x 2
##    Time           N
##  * <Duration> <int>
##  1 0s             7
##  2 2s             5
##  3 3s             4
##  4 4s             5
##  5 5s             4
##  6 6s             2
##  7 7s             2
##  8 8s             5
##  9 9s             3
## 10 10s            3
## # ... with 1,593 more rows
```

```
# CPET_rare %>% as_tsibble(index = Time, key = PatientId)
# The above line throws an error: Says there are duplicates.
# Check for duplicates:
(CPET_rare %>%
  duplicates(index = Time, key = PatientId) -> CPET_dupes)
```

```
## # A tibble: 4 x 17
##   PatientId Time                 TestLevel    HR   VO2 `VO2/kg`  VCO2    RQ
##       <int> <Duration>           <fct>     <dbl> <dbl>    <dbl> <dbl> <dbl>
## 1        47 435s (~7.25 minutes) Exercise    136  1.64     15.9  1.72    NA
## 2        47 435s (~7.25 minutes) Exercise    136  1.12     10.8  1.09    NA
## 3        55 69s (~1.15 minutes)  Test        107  0.24      2.8  0.21    NA
## 4        55 69s (~1.15 minutes)  Test        107  0.36      4.1  0.31    NA
## # ... with 9 more variables: VE <dbl>, `VE/VO2` <dbl>, `VE/VCO2` <dbl>,
## #   Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>, TMSPD <dbl>,
## #   TMELV <dbl>
```

```
# Examine these duplicates in the original CPET_raw table
CPET_raw %>%
  filter(PatientId %in% CPET_dupes$PatientId & lubridate::as.duration(hms(Time)) %in% CPET_dupes$Time)
```

```
## # A tibble: 4 x 19
##   PatientId SessionId Time  TestLevel    HR SpO2    VO2 `VO2/kg`  VCO2    RQ
##   <chr>     <chr>     <chr> <chr>     <dbl> <chr> <dbl>    <dbl> <dbl> <dbl>
## 1 47        47        0:07~ Exercise    136 <NA>   1.64     15.9  1.72    NA
## 2 47        47        0:07~ Exercise    136 <NA>   1.12     10.8  1.09    NA
## 3 55        55        0:01~ Test        107 <NA>   0.24      2.8  0.21    NA
## 4 55        55        0:01~ Test        107 <NA>   0.36      4.1  0.31    NA
## # ... with 9 more variables: VE <dbl>, `VE/VO2` <dbl>, `VE/VCO2` <dbl>,
## #   Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>, TMSPD <dbl>,
## #   TMELV <dbl>
```

```
# Need somehow to resolve or drop duplicates.
# This code will drop the second record in each duplicated group:
```

```
CPET_dupes %>%
      group_by(PatientId, Time) %>%
      slice(1)
```

```
## # A tibble: 2 x 17
## # Groups:   PatientId, Time [2]
##   PatientId Time                 TestLevel    HR   VO2 `VO2/kg`  VCO2    RQ
##       <int> <Duration>           <fct>     <dbl> <dbl>    <dbl> <dbl> <dbl>
## 1        47 435s (~7.25 minutes) Exercise    136  1.64     15.9  1.72    NA
## 2        55 69s (~1.15 minutes)  Test        107  0.24      2.8  0.21    NA
## # ... with 9 more variables: VE <dbl>, `VE/VO2` <dbl>, `VE/VCO2` <dbl>,
## #   Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>, TMSPD <dbl>,
## #   TMELV <dbl>
```

```
CPET_rare %>%
      group_by(PatientId, Time) %>%
      slice(1) %>%
      ungroup() -> CPET_medium

# Reference: https://stackoverflow.com/questions/22959635/remove-duplicated-rows-using-dplyr
```

```
(CPET_medium %>% as_tsibble(index = Time, key = PatientId) -> CPET_tbl_ts)
```

```
## # A tsibble: 16,562 x 17 [1]
## # Key:       PatientId [179]
##    PatientId Time                 TestLevel    HR   VO2 `VO2/kg`  VCO2    RQ
##        <int> <Duration>           <fct>     <dbl> <dbl>    <dbl> <dbl> <dbl>
##  1         1 20s                  Baseline     74 0.601      6.4 0.5    0.84
##  2         1 40s                  Baseline     74 0.492      5.2 0.48   0.99
##  3         1 60s (~1 minutes)     Baseline     73 0.476      5   0.443  0.93
##  4         1 80s (~1.33 minutes)  Baseline     74 0.44       4.7 0.389  0.88
##  5         1 100s (~1.67 minutes) Baseline     75 0.452      4.8 0.433  0.98
##  6         1 120s (~2 minutes)    Baseline     74 0.467      4.9 0.4    0.85
##  7         1 140s (~2.33 minutes) Baseline     78 0.536      5.7 0.385  0.74
##  8         1 160s (~2.67 minutes) Exercise     86 0.808      8.6 0.531  0.66
##  9         1 180s (~3 minutes)    Exercise     86 0.696      7.4 0.475  0.68
## 10         1 200s (~3.33 minutes) Exercise     86 0.796      8.4 0.604  0.76
## # ... with 16,552 more rows, and 9 more variables: VE <dbl>, `VE/VO2` <dbl>,
## #   `VE/VCO2` <dbl>, Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>,
## #   TMSPD <dbl>, TMELV <dbl>
```

```
interval(CPET_tbl_ts)
```

```
## <interval[1]>
## [1] 1
```

```
is_regular(CPET_tbl_ts)
```

```
## [1] TRUE
```

```
is_ordered(CPET_tbl_ts)
```

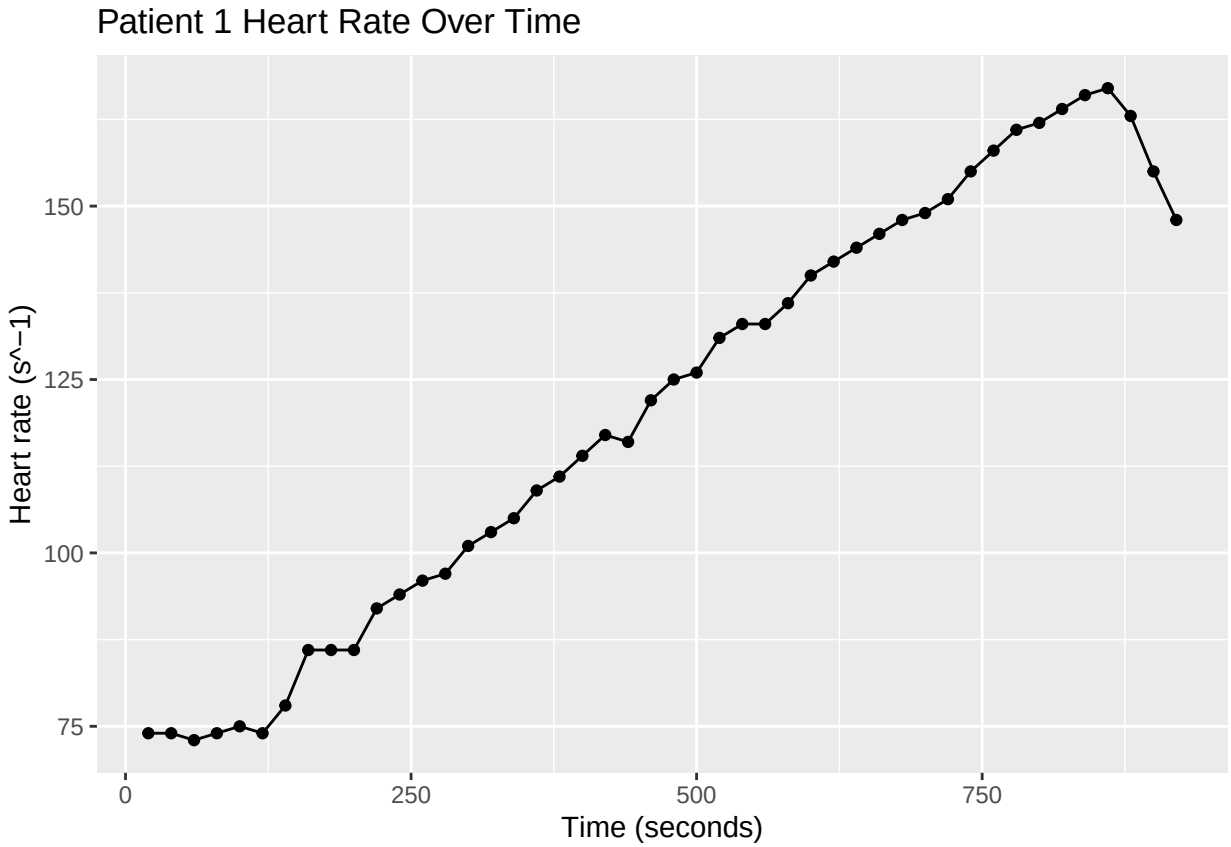
```
## [1] TRUE
```

```
has_gaps(CPET_tbl_ts)
```

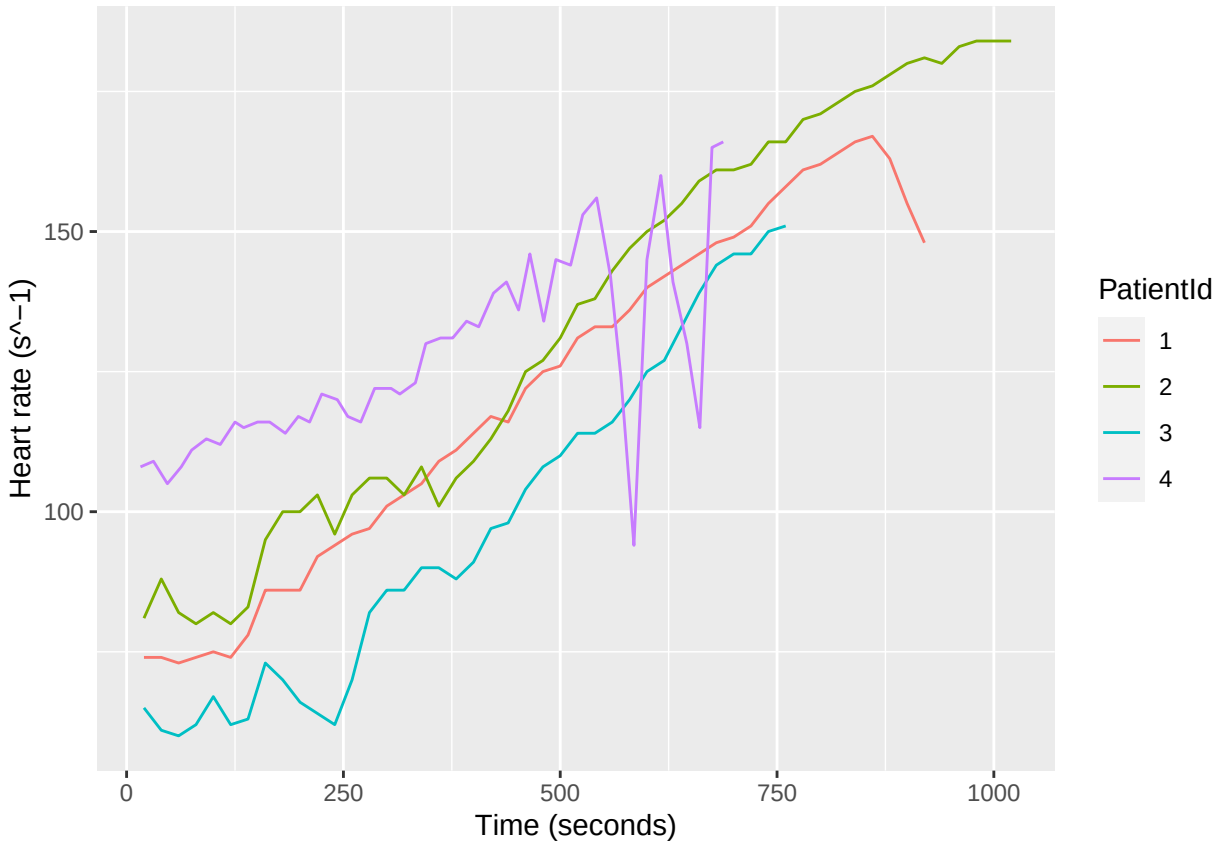
```
## # A tibble: 179 x 2
##    PatientId .gaps
```

```
##         <int> <lgl>
##  1          1 TRUE
##  2          2 TRUE
##  3          3 TRUE
##  4          4 TRUE
##  5          5 TRUE
##  6          6 TRUE
##  7          7 TRUE
##  8          8 TRUE
##  9          9 TRUE
## 10         10 TRUE
## # ... with 169 more rows
```

```
CPET_tbl_ts %>%
  filter(PatientId == 1) %>%
  ggplot(mapping = aes(x = Time, y = HR)) +
  geom_line() + geom_point() +
  ggtitle("Patient 1 Heart Rate Over Time") + xlab("Time (seconds)") + ylab("Heart rate (s^-1)")
```




```
CPET_tbl_ts %>%
  filter(PatientId %in% 1:4) %>%
  feasts::autoplot(HR) + xlab("Time (seconds)") + ylab("Heart rate (s^-1)")
```




```
# Reference: https://community.rstudio.com/t/how-to-bin-a-time-column-in-r/44867/2
CPET_medium %>%
  mutate(Time_20s = hms::round_hms(Time, secs = 20)) %>%
  group_by(PatientId, Time_20s) %>%
  summarise(HR20 = mean(HR)) %>%
  ungroup() %>%
  as_tsibble(index = Time_20s, key = PatientId)
```

```
## `summarise()` has grouped output by 'PatientId'. You can override using the `.groups` argument.
```

```
## # A tsibble: 6,411 x 3 [20]
## # Key:       PatientId [179]
##    PatientId Time_20s              HR20
##        <int> <Duration>           <dbl>
##  1         1 20s                    74
##  2         1 40s                    74
##  3         1 60s (~1 minutes)       73
##  4         1 80s (~1.33 minutes)    74
##  5         1 100s (~1.67 minutes)   75
##  6         1 120s (~2 minutes)      74
##  7         1 140s (~2.33 minutes)   78
##  8         1 160s (~2.67 minutes)   86
##  9         1 180s (~3 minutes)      86
## 10         1 200s (~3.33 minutes)   86
## # ... with 6,401 more rows
```

```
CPET_medium %>% filter(HR == 0)
```

```
## # A tibble: 153 x 17
##    PatientId Time                 TestLevel    HR   VO2 `VO2/kg`  VCO2    RQ
##        <int> <Duration>           <fct>     <dbl> <dbl>    <dbl> <dbl> <dbl>
##  1        16 17s                  Warmup        0 0.271     5.63 0.264    NA
##  2        16 31s                  Warmup        0 0.256     5.32 0.265    NA
##  3        16 47s                  Warmup        0 0.325     6.76 0.346    NA
##  4        16 60s (~1 minutes)     Warmup        0 0.472     9.81 0.464    NA
##  5        16 77s (~1.28 minutes)  Warmup        0 0.227     4.71 0.223    NA
##  6        16 91s (~1.52 minutes)  Warmup        0 0.213     4.42 0.219    NA
##  7        16 108s (~1.8 minutes)  Warmup        0 0.172     3.57 0.183    NA
##  8        16 126s (~2.1 minutes)  Warmup        0 0.201     4.18 0.214    NA
##  9        16 137s (~2.28 minutes) Warmup        0 0.291     6.06 0.302    NA
## 10        16 151s (~2.52 minutes) Exercise      0 0.213     4.43 0.211    NA
## # ... with 143 more rows, and 9 more variables: VE <dbl>, `VE/VO2` <dbl>,
## #   `VE/VCO2` <dbl>, Work <dbl>, PetO2 <dbl>, PetCO2 <dbl>, VEO22 <dbl>,
## #   TMSPD <dbl>, TMELV <dbl>
```

It looks like some of the patients have a recorded heart rate of zero. Either there is another round of data cleaning to do, or some other procedue should be followed.