tidyな時系列解析

20190302
Tokyo.R LT
@flaty13

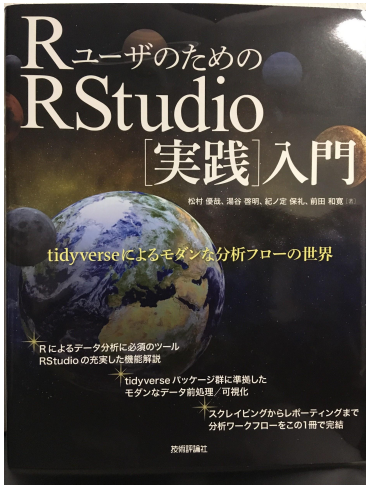
自己紹介

- twitter: @flaty13
- 普段扱っているデータ:
  - お仕事: ネットワークや機器のログなど
  - 息抜き: スポーツ系（テニス、野球など）
- R歴: 2年くらい
  - 2017.4に入社して以来
  - 学部時代の講義で使ったらしいが何も覚えてないからノーカン（~~レポートのグラフをエクセルで作った形跡があった…~~）

tidy?

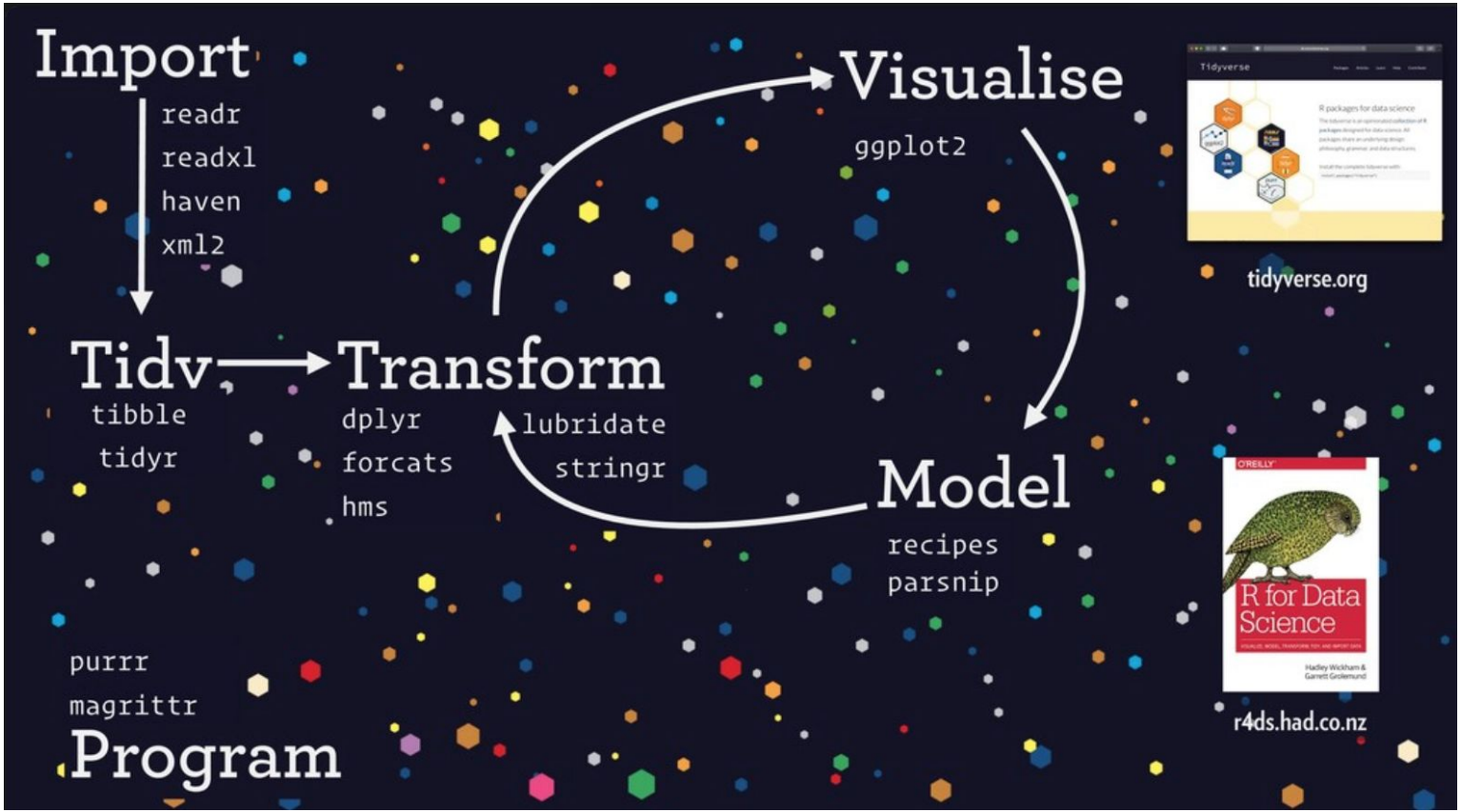
- Rを1年くらい使った頃の感想
  - 基本dplyrとggplot2の2トップ
  - tidyrは正直あんまり...
  - なんでdplyrやggplot2を差し置いてtidyrが主役みたいな名前なのん? > tidyverse

tidy?



- 宇宙本に書いてあった
- tidy dataの定義
  - 1つの列が1つの変数を表す
  - 1つの行が1つの観測を表す
  - 1つのテーブルが1つのデータセットだけを含む
- tidyrはデータをtidy dataにするのに便利なパッケージ

tidy?



https://speakerdeck.com/hadley/welcome-to-the-tidyverse?slide=29

tsibble

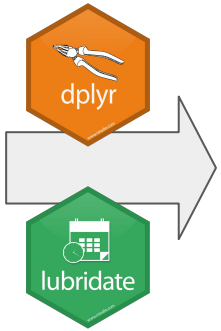
- 時系列データに関して、tidyverseのパッケージ群では不便な場面があったりする

timestamp	id	value
2019-01-01 09:51:43	A	20
2019-01-01 09:54:22	B	5
2019-01-01 09:55:40	A	120
2019-01-01 09:58:01	B	40
2019-01-01 10:01:12	A	60
2019-01-01 10:02:59	A	80
⋮		

tsibble

- 時系列データに関して、tidyverseのパッケージ群では不便なこともあったりする

timestamp	id	value
2019-01-01 09:51:43	A	20
2019-01-01 09:54:22	B	5
2019-01-01 09:55:40	A	120
2019-01-01 09:58:01	B	40
2019-01-01 10:01:12	A	60
2019-01-01 10:02:59	A	80
⋮		



timestamp	id	value
2019-01-01 09:50:00	A	2010
2019-01-01 09:50:00	B	1300
2019-01-01 10:00:00	A	2600
2019-01-01 10:10:00	A	1500
2019-01-01 10:10:00	B	120
⋮		

tsibble

- 時系列データに関して、tidyverseのパッケージ群では不便なこともあったりする

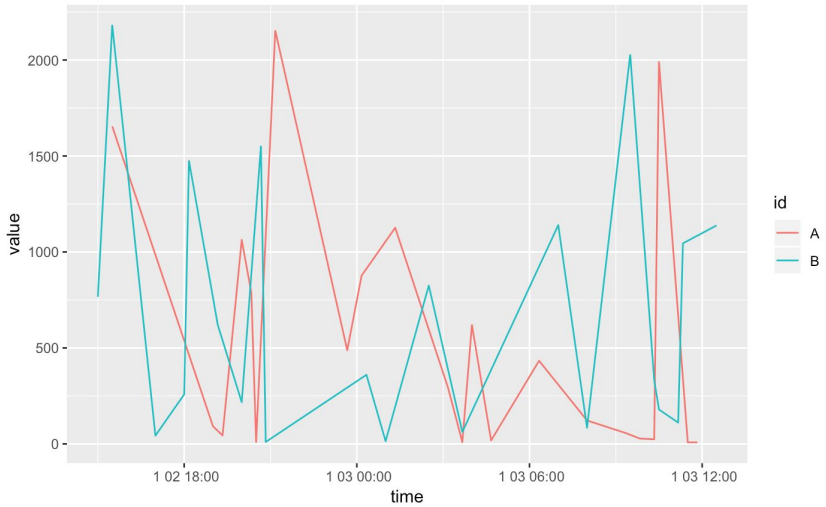
timestamp	id	value
2019-01-01 09:51:43	A	20
2019-01-01 09:54:22	B	5
2019-01-01 09:55:40	A	120
2019-01-01 09:58:01	B	40
2019-01-01 10:01:12	A	60
2019-01-01 10:02:59	A	80
⋮		

dplyr

lubridate

timestamp	id	value
2019-01-01 09:50:00	A	2010
2019-01-01 09:50:00	B	1300
2019-01-01 10:00:00	A	2600
2019-01-01 10:10:00	A	1500
2019-01-01 10:10:00	B	120
⋮		

ggplot2



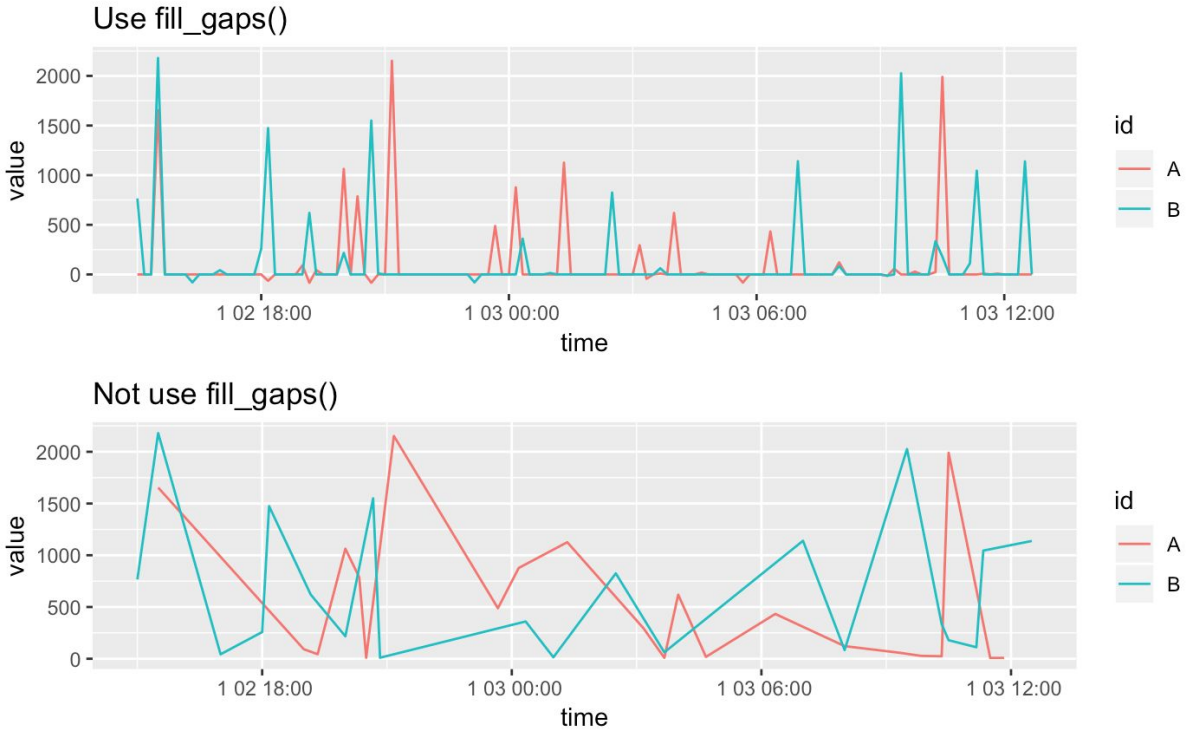
- 単時系列ならtidyr::complete()で補間できる
- 複数時系列だとうまくいかない

tsibble

- tsibble::fill_gaps()で複数時系列の補間も一発でできる

```
tsibble::fill_gaps(value = 0) %>%
```

↑この1行を入れるだけでOK



（一旦）まとめ

- tsibble便利
- （時系列補間以外にも便利な機能がある）
- 時系列データを扱う人は是非！
- 開発者Earo Wangさんのドキュメントがわかりやすいので
  詳しくはそちらを！
  - Reintroducing tsibble: data tools that melt the clock
  - MELT THE CLOCK

歩数のデータに使ってみる

- iPhoneが(いつの間にか)取得していた歩数のデータ
  - 今回は2019年1月以降でfilter
  - データ自体は2016年7月からあった...

歩数のデータに使ってみる

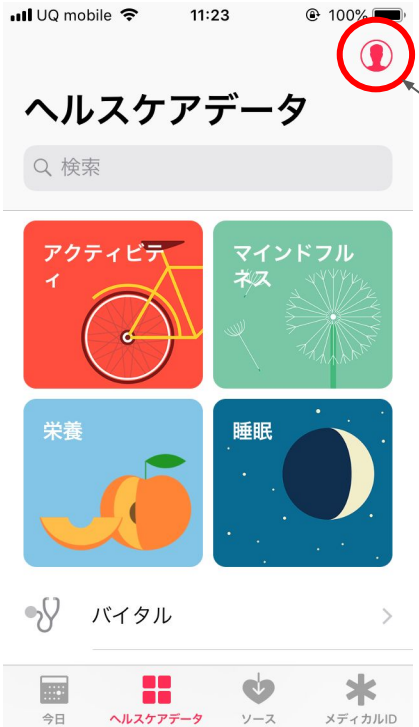
- やったこと
  - iPhoneのヘルスケアアプリからデータをダウンロード
  - xml形式のデータから必要な部分だけ抽出
  - 1時間あたりの合計で集計
  - tsibble化
  - fill_gaps()で補間 & sugrrantsでカレンダープロット

歩数のデータに使ってみる

- やったこと
  - iPhoneのヘルスケアアプリからデータをダウンロード
  - xml形式のデータから必要な部分だけ抽出
  - 1時間あたりの合計で集計
  - as_tsibble()
  - fill_gaps()で補間 & sugrrantsでカレンダープロット

歩数のデータに使ってみる

- iPhoneのヘルスケアアプリからデータをダウンロード



ここをクリック

ここからデータを書き出せる

歩数のデータに使ってみる

- xml形式のデータから必要な部分だけ抽出

```
#-----iPhoneからDLしたxmlデータの読み込み&加工-----

#Record部分を抽出
tmp <- XML::xmlParse("healthcare.xml")  %>%
  XML::getNodeSet("//Record")

#今回の分析に使うstartDateとvalueを抽出
time <- tmp %>% sapply(function(x) xmlGetAttr(x, "startDate"))
value <- tmp %>% sapply(function(x) xmlGetAttr(x, "value"))

#tibble化&期間指定&id追加
df <- tibble(time, value) %>%
  dplyr::filter(time >= "2019-01-01 09:00:00") %>%
  dplyr::mutate(id = "flaty")
```

今回はhealthcare.xml
という名前のファイルに
してRで読み込んだ

歩数のデータに使ってみる

- xml形式のデータから必要な部分だけ抽出

```
## # A tibble: 2,645 x 3
##    time                value id
##    <dttm>              <int> <chr>
##  1 2019-01-01 09:51:43    42 flaty
##  2 2019-01-01 15:23:56    64 flaty
##  3 2019-01-01 17:52:49    66 flaty
##  4 2019-01-01 18:03:19    28 flaty
##  5 2019-01-01 18:24:51    14 flaty
##  6 2019-01-01 18:25:06     3 flaty
##  7 2019-01-02 08:51:37    32 flaty
##  8 2019-01-02 09:23:57    14 flaty
##  9 2019-01-02 09:52:39    36 flaty
## 10 2019-01-02 11:38:34    38 flaty
## # … with 2,635 more rows
```

抽出したデータはこんな感じ

- time: 歩き始めた時間
- value: 歩数
- id: tsibble化するために付加
    - 今回はflatyのみ
    - 複数の人のデータの場合はここに入るイメージ

歩数のデータに使ってみる

- 1時間あたりの合計で集計(dplyr + lubridate)

```
df %>%
  dplyr::group_by(time = lubridate::floor_date(time, unit = "1 hour"), id) %>%
  dplyr::summarise(value = sum(value))
```

```
## # A tibble: 559 x 3
## # Groups:   time [559]
##    time                id    value
##    <dttm>              <chr> <int>
##  1 2019-01-01 09:00:00 flaty    42
##  2 2019-01-01 15:00:00 flaty    64
##  3 2019-01-01 17:00:00 flaty    66
##  4 2019-01-01 18:00:00 flaty    45
##  5 2019-01-02 08:00:00 flaty    32
##  6 2019-01-02 09:00:00 flaty    50
##  7 2019-01-02 11:00:00 flaty    38
##  8 2019-01-02 13:00:00 flaty    14
##  9 2019-01-02 14:00:00 flaty  4916
## 10 2019-01-02 15:00:00 flaty  1975
## # … with 549 more rows
```

歩数のデータに使ってみる

- tsibble化

```
df_ts <- df %>%
  tsibble::as_tsibble(key = id(id), index = time)
```

```
## # A tsibble: 559 x 3 [1h] <UTC>
## # Key:       id [1]
## # Groups:    @ time [559]
##    time                id    value
##    <dttm>              <chr> <int>
##  1 2019-01-01 09:00:00 flaty    42
##  2 2019-01-01 15:00:00 flaty    64
##  3 2019-01-01 17:00:00 flaty    66
##  4 2019-01-01 18:00:00 flaty    45
##  5 2019-01-02 08:00:00 flaty    32
##  6 2019-01-02 09:00:00 flaty    50
##  7 2019-01-02 11:00:00 flaty    38
##  8 2019-01-02 13:00:00 flaty    14
##  9 2019-01-02 14:00:00 flaty  4916
## 10 2019-01-02 15:00:00 flaty  1975
## # … with 549 more rows
```

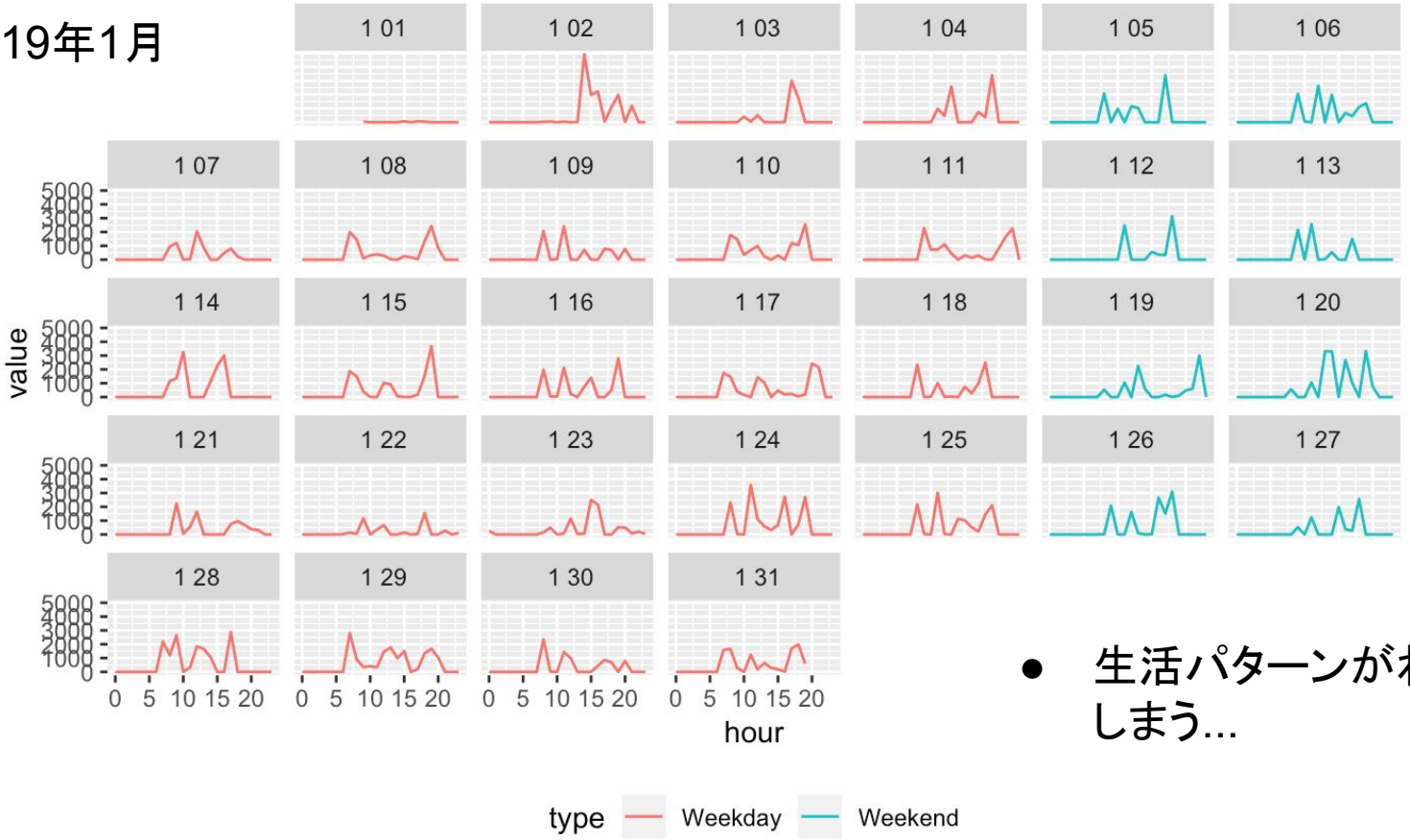
歩数のデータに使ってみる

- sugrrantsでカレンダープロット
  - tsibbleと同じくEaro Wangさんが開発したsuggrantというパッケージでplotする

```
df_ts %>%
  dplyr::filter(time >= "2019-01-01 09:00:00" & time < "2019-02-01 09:00:00") %>%
  tsibble::fill_gaps(value = 0) %>%
  dplyr::mutate(date = as.Date(time), day = weekdays(time), hour = hour(time)) %>%
  dplyr::mutate(type = if_else(day %in% c("土曜日", "日曜日"), "Weekend", "Weekday")) %>%
  ggplot(aes(x = hour, y = value, group = date, colour = type)) +
  geom_line() +
  sugrrants::facet_calendar(~ date) +
  theme(legend.position = "bottom")
```

歩数のデータに使ってみる

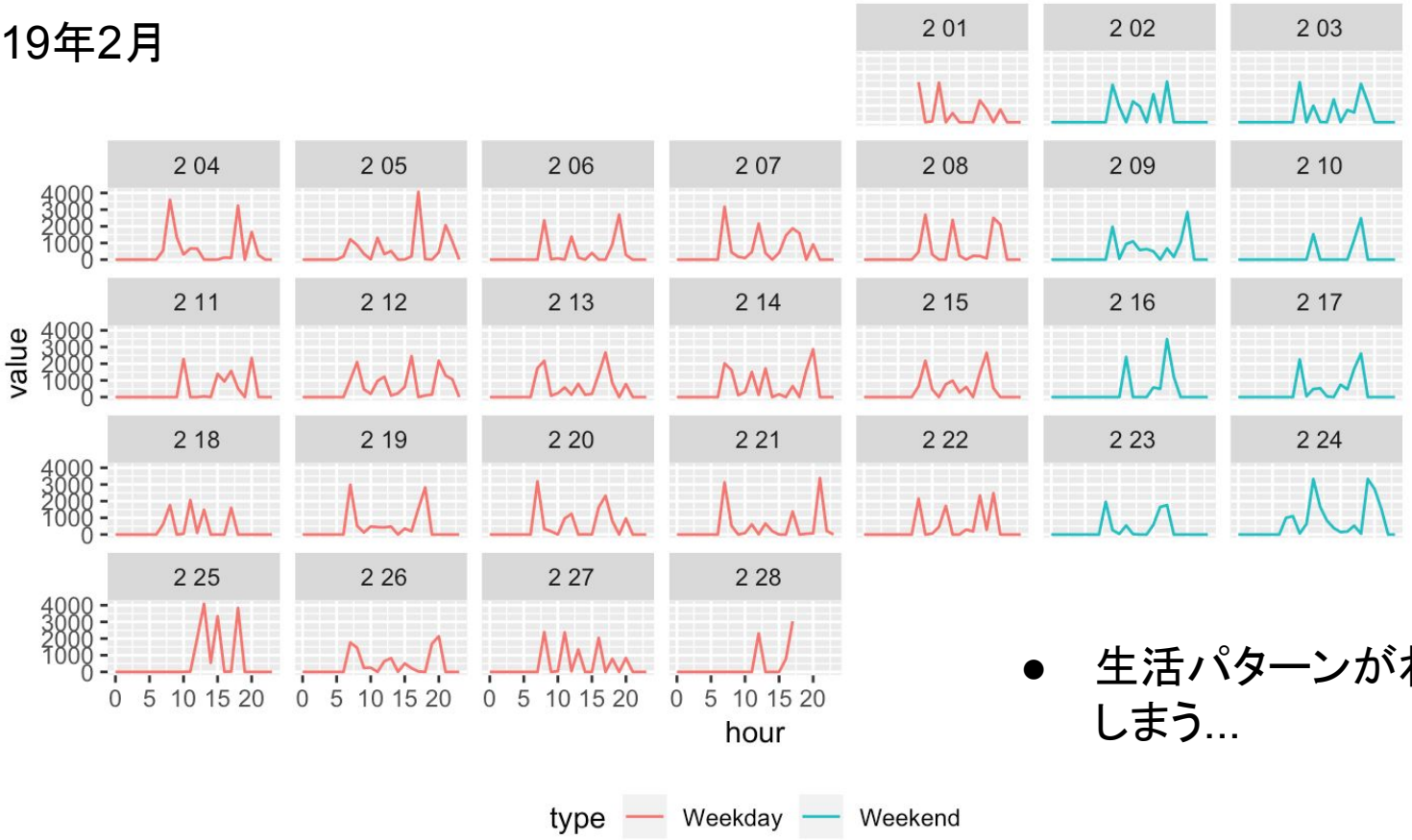
2019年1月



- 生活パターンがわかってしまう…

歩数のデータに使ってみる

2019年2月



- 生活パターンがわかってしまう…

Reference

- 宇宙本
- R for Data Science - Tidy data
- welcome to the tidyverse
- Reintroducing tsibble: data tools that melt the clock
- MELT THE CLOCK
- Earo Wangさんのページ

Enjoy!