

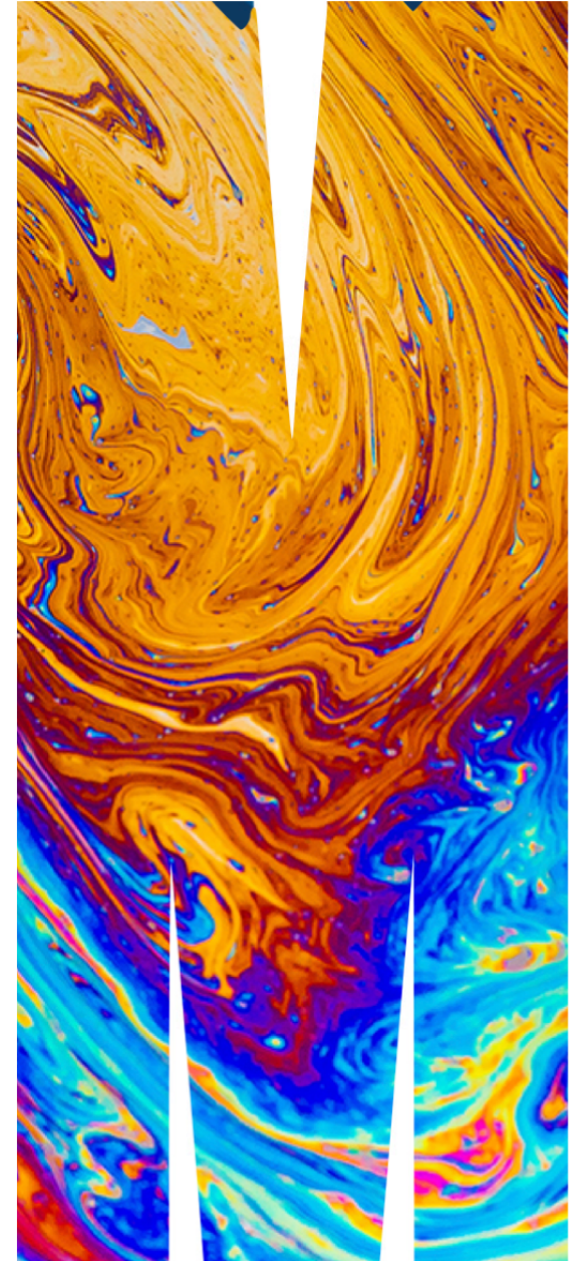
# ETC5521: Exploratory Data Analysis

**Exploring data having a space and time context**



Lecturer: *Di Cook*

✉ ETC5521.Clayton-x@monash.edu

📅 Week 9 - Session 2



# Outline

-  temporal missing values: time series models require that there is a value for each time step
-  longitudinal data: it is different from time series. Typically measurements are taken irregularly in time.

# Working with missings

# Checking counting and filling missings in time

```
set.seed(328)
harvest <- tsibble(
  year = c(2010, 2011, 2013, 2011,
           2012, 2013),
  fruit = rep(c("kiwi", "cherry"),
              each = 3),
  kilo = sample(1:10, size = 6),
  key = fruit, index = year
)
harvest
```

```
## # A tsibble: 6 x 3 [1Y]
## # Key:      fruit [2]
##   year fruit  kilo
##   <dbl> <chr> <int>
## 1  2011 cherry     2
## 2  2012 cherry     7
## 3  2013 cherry     1
## 4  2010 kiwi      6
## 5  2011 kiwi      5
## 6  2013 kiwi      8
```

```
has_gaps(harvest, .full = TRUE)
```

```
## # A tibble: 2 x 2
##   fruit .gaps
##   <chr> <lgl>
## 1 cherry TRUE
## 2 kiwi  TRUE
```

Both levels of the key have missings.

Can you see the gaps in time?

# Checking counting and filling missings in time

```
set.seed(328)
harvest <- tsibble(
  year = c(2010, 2011, 2013, 2011,
           2012, 2013),
  fruit = rep(c("kiwi", "cherry"),
             each = 3),
  kilo = sample(1:10, size = 6),
  key = fruit, index = year
)
harvest
```

```
## # A tsibble: 6 x 3 [1Y]
## # Key:      fruit [2]
##   year fruit  kilo
##   <dbl> <chr> <int>
## 1  2011 cherry     2
## 2  2012 cherry     7
## 3  2013 cherry     1
## 4  2010 kiwi      6
## 5  2011 kiwi      5
## 6  2013 kiwi      8
```

```
count_gaps(harvest, .full=TRUE)
```

```
## # A tibble: 2 x 4
##   fruit .from .to .n
##   <chr> <dbl> <dbl> <int>
## 1 cherry  2010  2010     1
## 2 kiwi    2012  2012     1
```

One missing in each level, although it is a different year.

Notice how `tsibble` handles this summary so neatly.

# Checking counting and filling missings in time

```
set.seed(328)
harvest <- tsibble(
  year = c(2010, 2011, 2013, 2011,
           2012, 2013),
  fruit = rep(c("kiwi", "cherry"),
             each = 3),
  kilo = sample(1:10, size = 6),
  key = fruit, index = year
)
harvest

## # A tsibble: 6 x 3 [1Y]
## # Key:      fruit [2]
##   year fruit  kilo
##   <dbl> <chr> <int>
## 1  2011 cherry     2
## 2  2012 cherry     7
## 3  2013 cherry     1
## 4  2010 kiwi      6
## 5  2011 kiwi      5
## 6  2013 kiwi      8
```

Make the implicit missing values **explicit**.

```
harvest <- fill_gaps(harvest,
                    .full=TRUE)
harvest

## # A tsibble: 8 x 3 [1Y]
## # Key:      fruit [2]
##   year fruit  kilo
##   <dbl> <chr> <int>
## 1  2010 cherry    NA
## 2  2011 cherry     2
## 3  2012 cherry     7
## 4  2013 cherry     1
## 5  2010 kiwi      6
## 6  2011 kiwi      5
## 7  2012 kiwi     NA
## 8  2013 kiwi      8
```

# Checking counting and filling missings in time

```
set.seed(328)
harvest <- tsibble(
  year = c(2010, 2011, 2013, 2011,
           2012, 2013),
  fruit = rep(c("kiwi", "cherry"),
             each = 3),
  kilo = sample(1:10, size = 6),
  key = fruit, index = year
)
harvest
```

```
harvest_nomiss <- harvest %>%
  group_by(fruit) %>%
  mutate(kilo =
    na_interpolation(kilo)) %>%
  ungroup()
harvest_nomiss
```

```
## # A tsibble: 8 x 3 [1Y]
## # Key:      fruit [2]
##   year fruit  kilo
##   <dbl> <chr> <dbl>
## 1  2010 cherry    2
## 2  2011 cherry    2
## 3  2012 cherry    7
## 4  2013 cherry    1
## 5  2010 kiwi     6
## 6  2011 kiwi     5
## 7  2012 kiwi     6.5
## 8  2013 kiwi     8
```

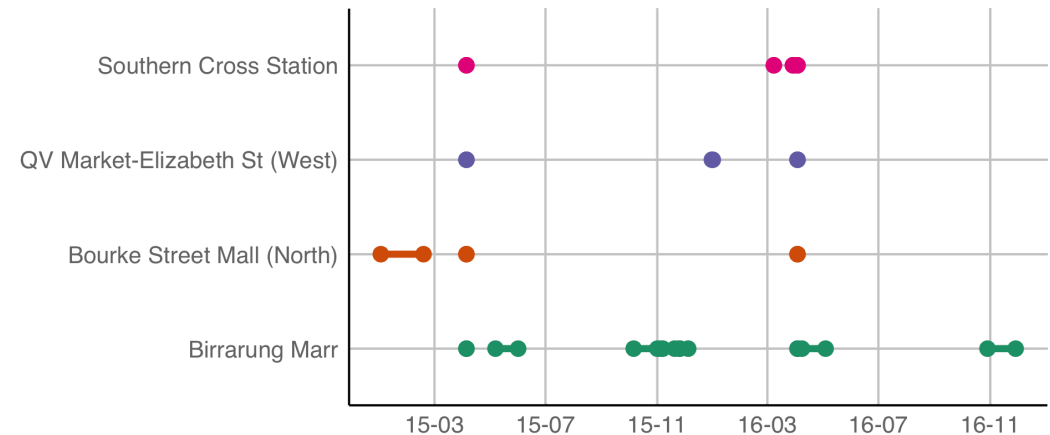
## Case study 3 Melbourne pedestrian traffic Part 1/5

```
data(pedestrian) # in tsibble
has_gaps(pedestrian,
        .full = TRUE)

## # A tibble: 4 × 2
##   Sensor                                .gaps
##   <chr>                                <lgl>
## 1 Birrarung Marr                      TRUE
## 2 Bourke Street Mall (North)         TRUE
## 3 QV Market-Elizabeth St (West)     TRUE
## 4 Southern Cross Station             TRUE

ped_gaps <- pedestrian %>%
  count_gaps(.full = TRUE)

ggplot(ped_gaps,
       aes(x = Sensor,
           colour = Sensor)) +
  geom_linerange(
    aes(ymin = .from,
        ymax = .to), size=2) +
  geom_point(aes(y = .from), size=4) +
  geom_point(aes(y = .to), size=4) +
```



Every sensor has a missing value each April. **What happens in April each year?**

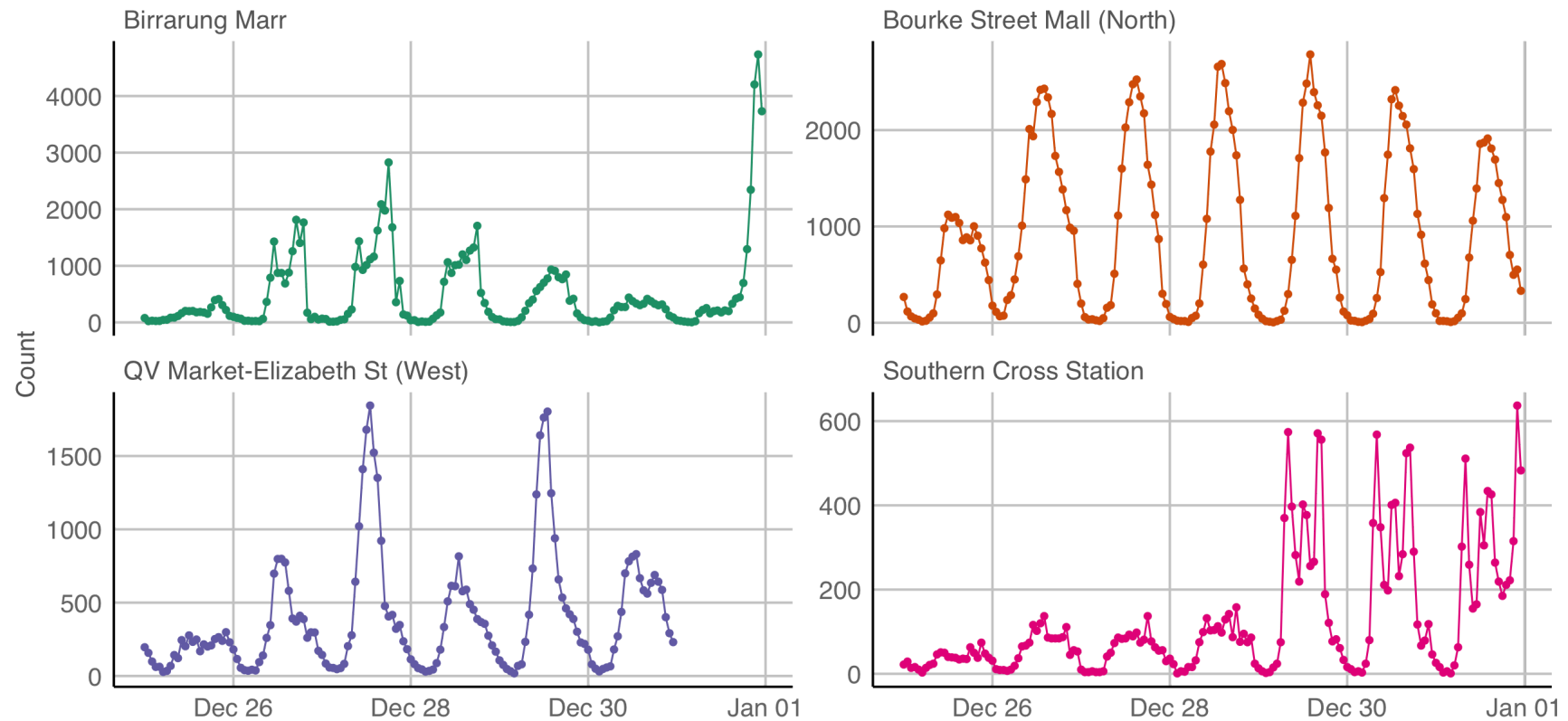


## Case study 3 Melbourne pedestrian traffic Part 2/5



R

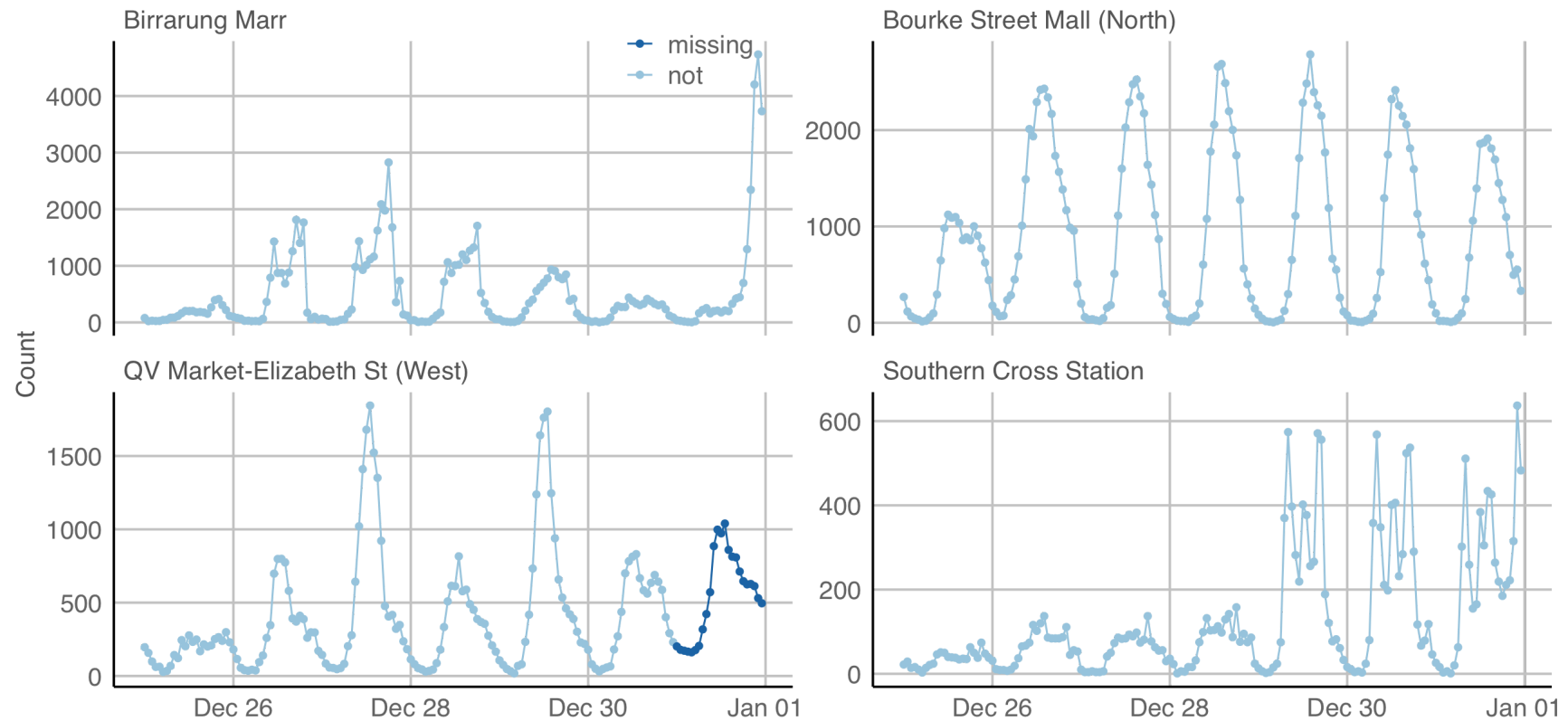
Missings at the end of the year at QV market.



## Case study 3 Melbourne pedestrian traffic Part 3/5



R

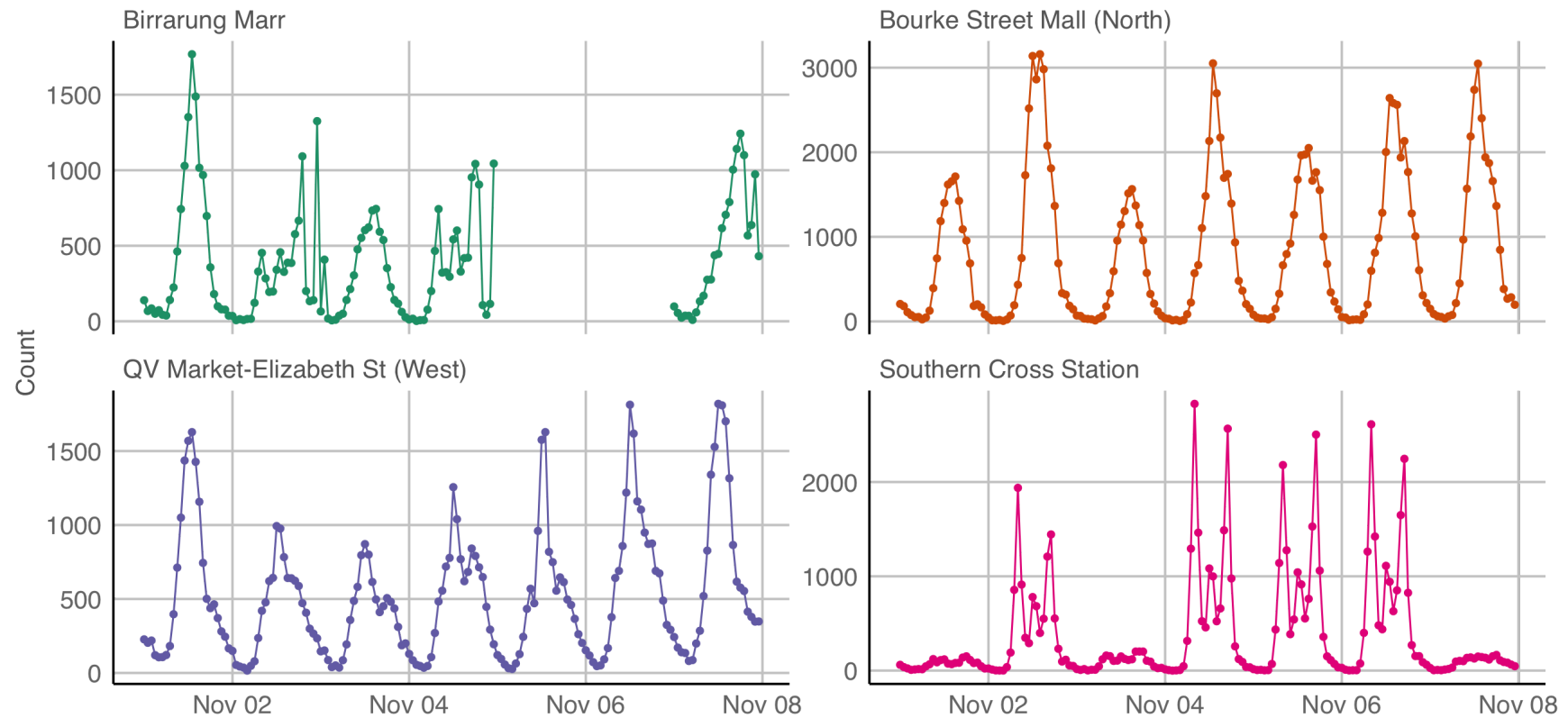


Imputed with seasonal component.

## Case study 3 Melbourne pedestrian traffic Part 4/5



R

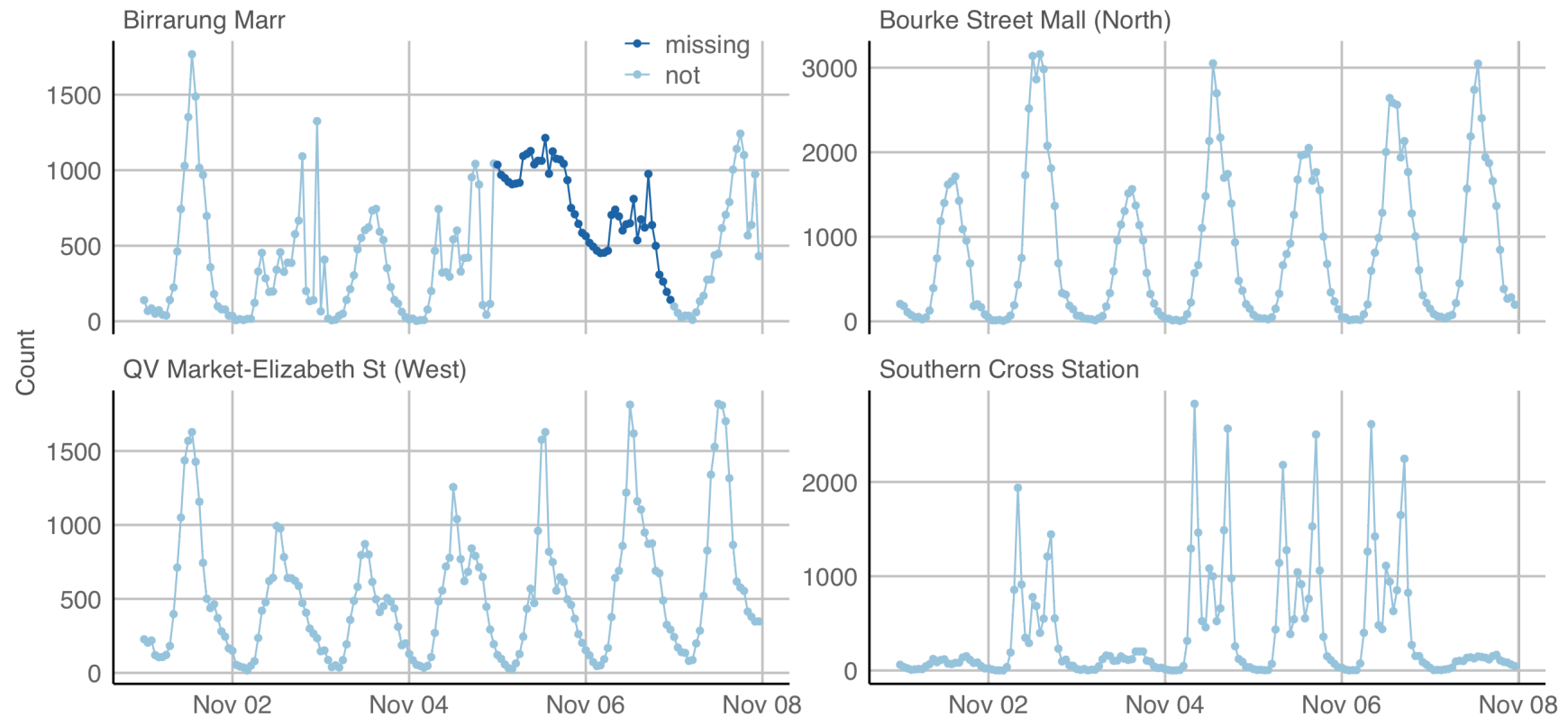


Missings in November at Birrarung Marr.

## Case study 3 Melbourne pedestrian traffic Part 5/5



R



Imputed with seasonal component. Irregular patterns make imputation difficult.



Imputing temporal data is necessary for modeling and forecasting, which typically require complete data. Incorporate seasonal components, if necessary, and temporal dependence. That means you need to **understand enough about the data to do imputation well.**

## **Longitudinal data**

Information from the same individuals, recorded at multiple points in time.

Usually irregular, and not easy to regularise. Lots more short series.

Longitudinal data has the same properties as time series, but generally different objectives for the analysis.

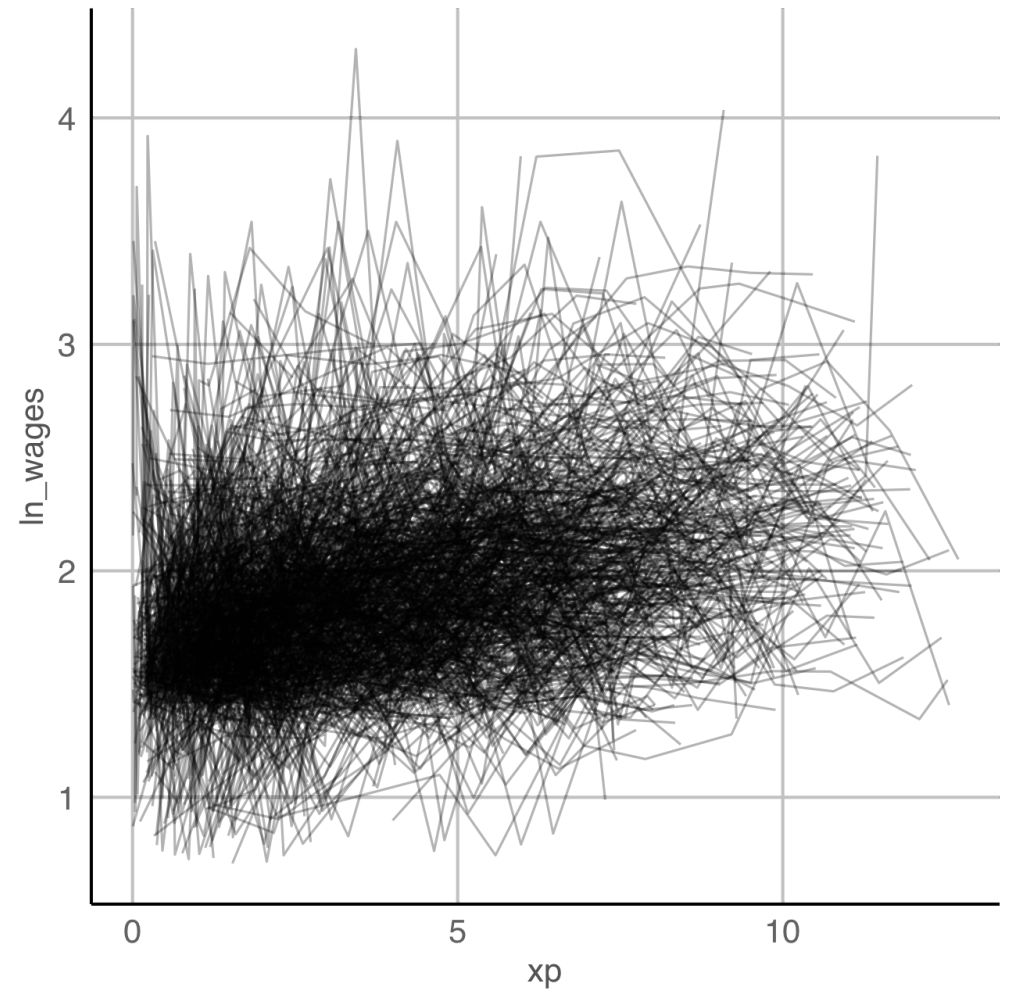
In the `brlgar` package methods build on the `tsibble` data object.

```
## # A tsibble: 6,402 x 9 [!]  
## # Key:           id [888]  
##       id ln_wages      xp    ged xp_since_ged black hispanic high_grade unemploy  
##   <int>   <dbl> <dbl> <int>         <dbl> <int>    <int>      <int>  
## 1     31     1.49 0.015     1         0.015     0        1         8  
## 2     31     1.43 0.715     1         0.715     0        1         8  
## 3     31     1.47 1.73      1         1.73      0        1         8  
## 4     31     1.75 2.77      1         2.77      0        1         8  
## 5     31     1.93 3.93      1         3.93      0        1         8  
## 6     31     1.71 4.95      1         4.95      0        1         8  
## 7     31     2.09 5.96      1         5.96      0        1         8  
## 8     31     2.13 6.98      1         6.98      0        1         8  
## 9     36     1.98 0.315     1         0.315     0        0         9  
## 10    36     1.80 0.983     1         0.983     0        0         9  
## # i 6,392 more rows
```

## Case study 4 Wages Part 1/15

```
wages %>%  
  ggplot(aes(x = xp,  
             y = ln_wages,  
             group = id)) +  
  geom_line(alpha=0.3)
```

Log(wages) of 888 individuals, measured at various times in their employment (workforce experience).





**to perfection**



Source: giphy

## Case study 4 Wages Part 2/15

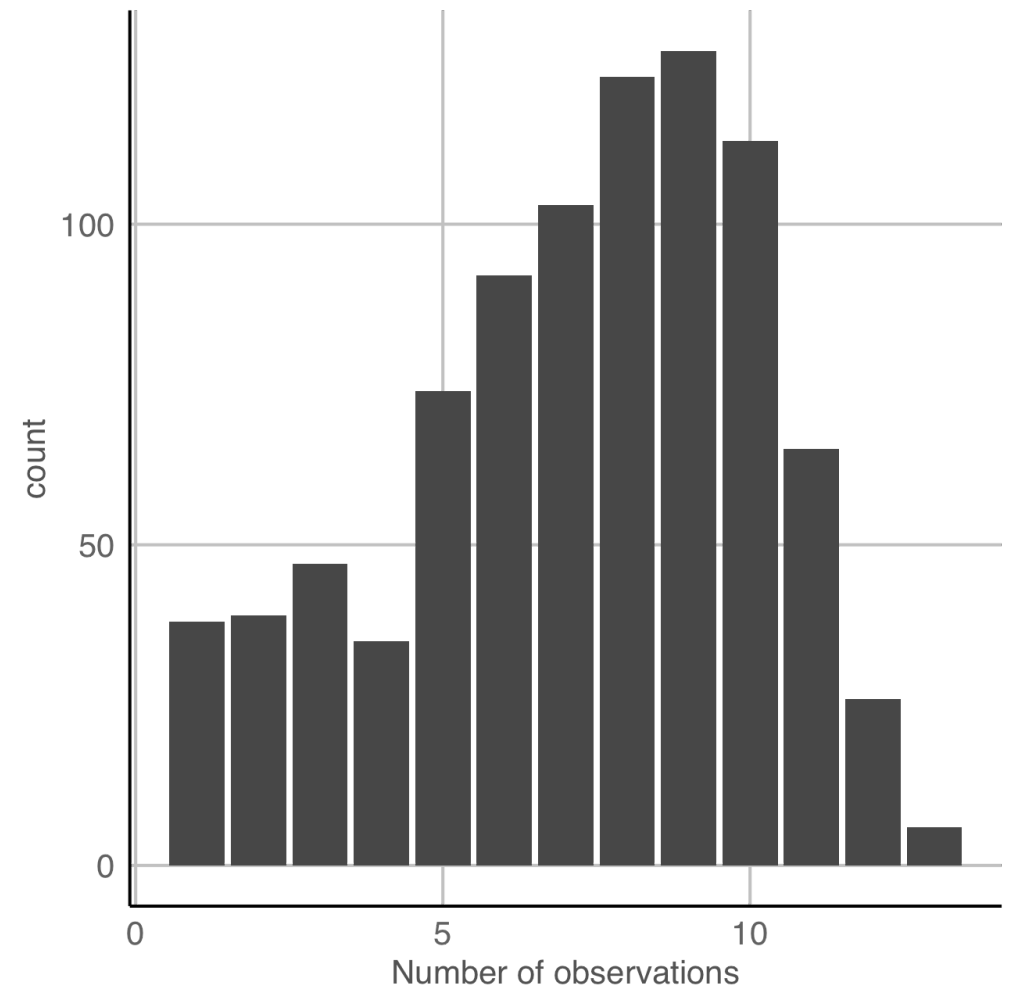
Using features, compute the number of measurements for each subject

```
wages %>%  
  features(ln_wages, n_obs) %>%  
  ggplot(aes(x = n_obs)) +  
  geom_bar() +  
  xlab("Number of observations")
```

Different number of observations per person! It ranges from 1-13.



Too few observations means there is a lack of support to do temporal analysis.



## Case study 4 Wages Part 3/15

You should filter on this, and remove subjects with few observations.

```
wages <- wages %>% add_n_obs()  
wages %>%  
  filter(n_obs > 3) %>%  
  select(id, ln_wages, xp, n_obs)
```

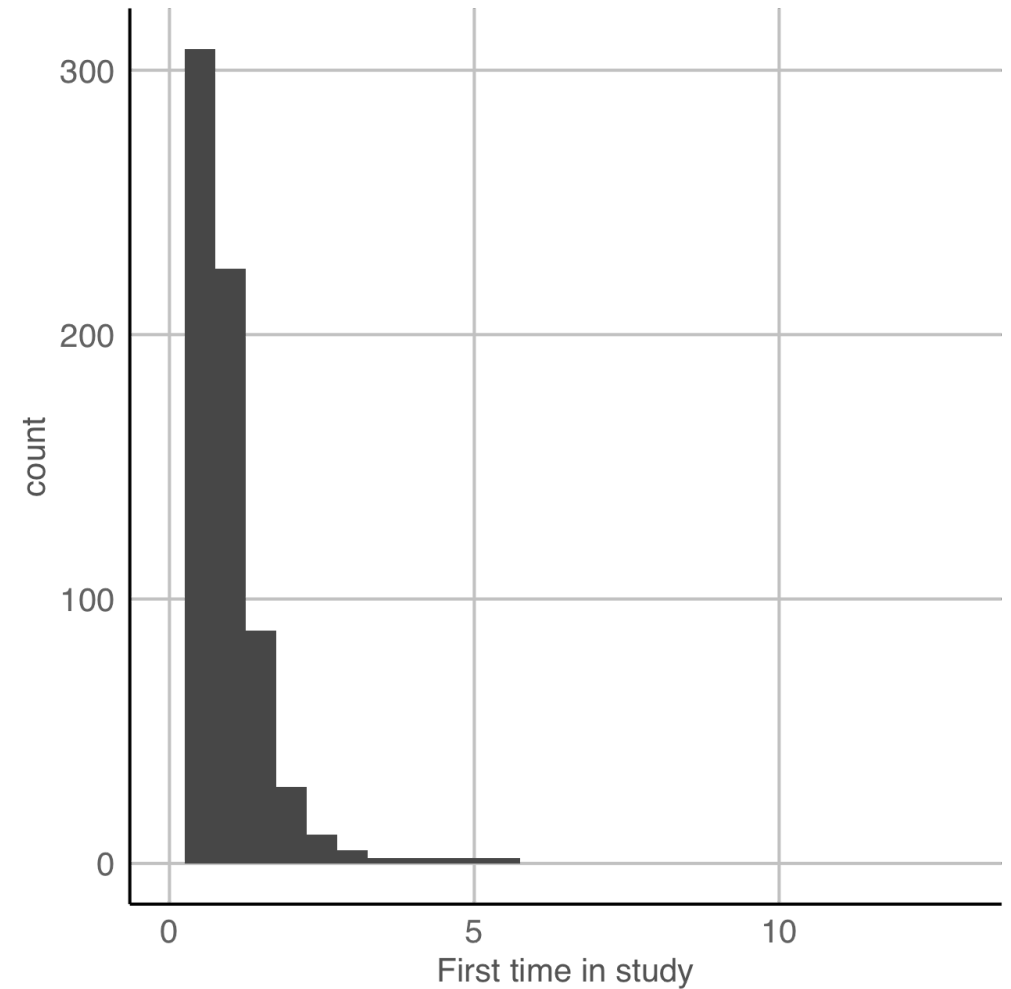
```
## # A tsibble: 6,145 x 4 [!]  
## # Key:      id [764]  
##       id ln_wages    xp n_obs  
##   <int>   <dbl> <dbl> <int>  
## 1     31    1.49 0.015     8  
## 2     31    1.43 0.715     8  
## 3     31    1.47 1.73     8  
## 4     31    1.75 2.77     8  
## 5     31    1.93 3.93     8  
## 6     31    1.71 4.95     8  
## 7     31    2.09 5.96     8  
## 8     31    2.13 6.98     8  
## 9     36    1.98 0.315    10  
## 10    36    1.80 0.983    10  
## # i 6,135 more rows
```

## Case study 4 Wages Part 4/15

Using features to extract minimum time

```
wages %>%  
  features(xp, list(min = min)) %>%  
  ggplot(aes(x = min)) +  
  geom_histogram(binwidth=0.5) +  
  xlim(c(0, 13)) +  
  xlab("First time in study")
```

Subjects start in the study at different employment experience times, ranging from 0 to more than 10 years.

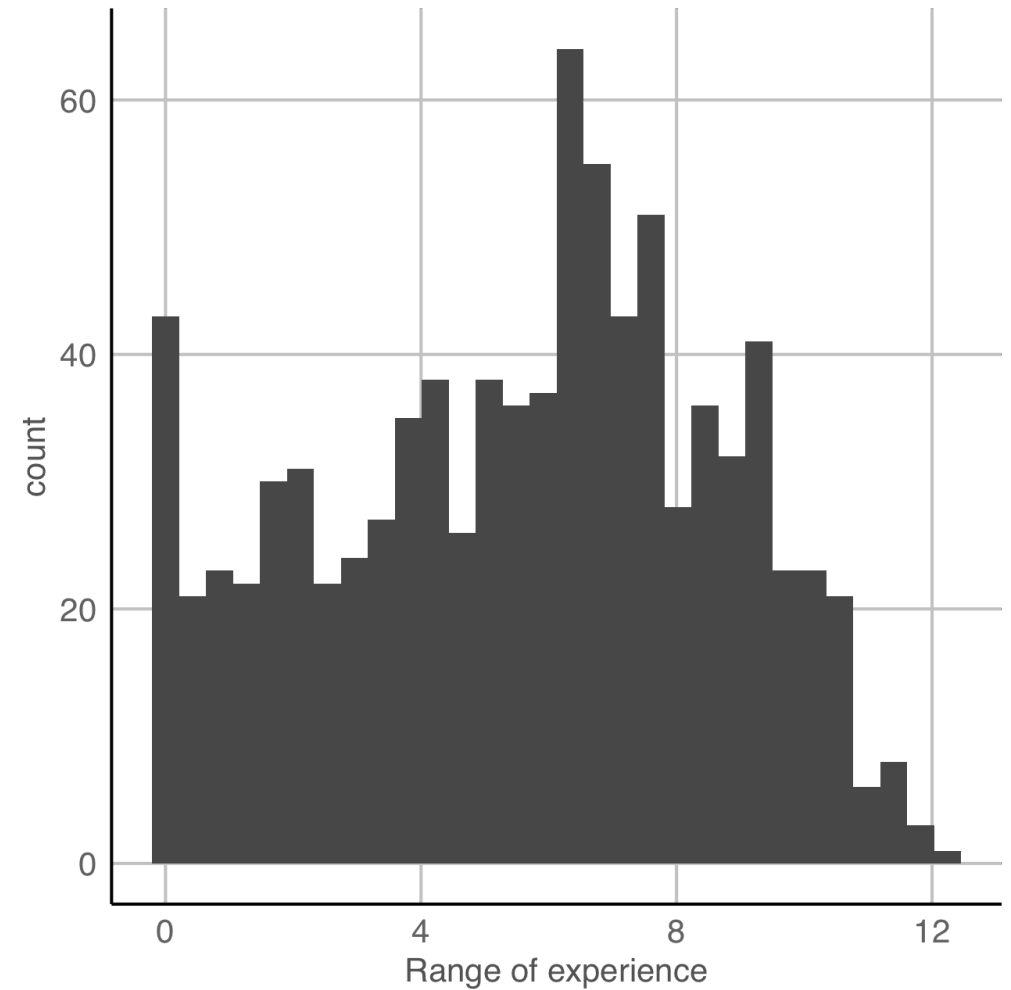


## Case study 4 Wages Part 5/15

Using features to extract range of time index

```
wages_xp_range <- wages %>%  
  features(xp, feat_ranges)  
  
ggplot(wages_xp_range,  
       aes(x = range_diff)) +  
  geom_histogram() +  
  xlab("Range of experience")
```

There's a range of workforce experience.



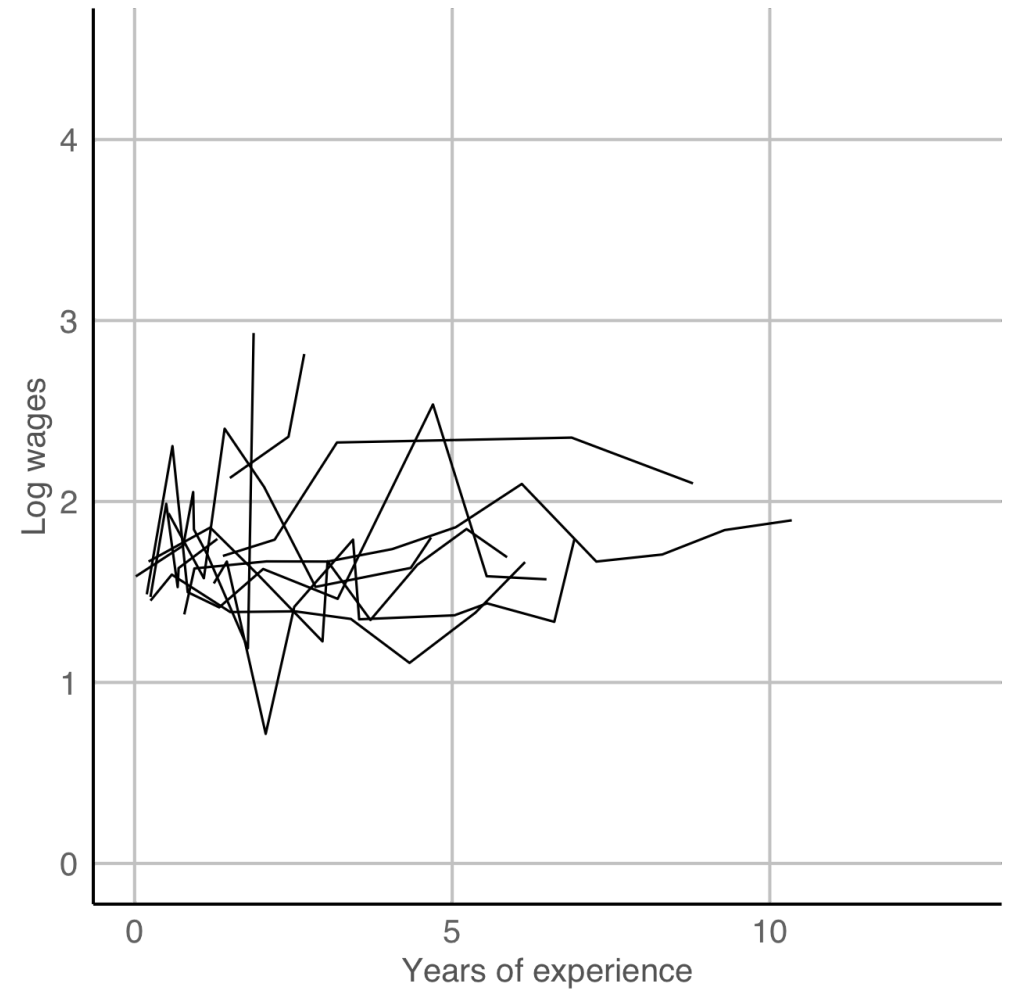
## Case study 4 Wages Part 6/15

Small spoonfuls of spaghetti

Sample some individuals

```
wages %>%  
  sample_n_keys(size = 10) %>%  
  ggplot(aes(x = xp,  
             y = ln_wages,  
             group = id)) +  
  geom_line() +  
  xlim(c(0, 13)) + ylim(c(0, 4.5)) +  
  xlab("Years of experience") +  
  ylab("Log wages")
```

Wages conversion 0.5 = \$1.65; 4.5 = \$90



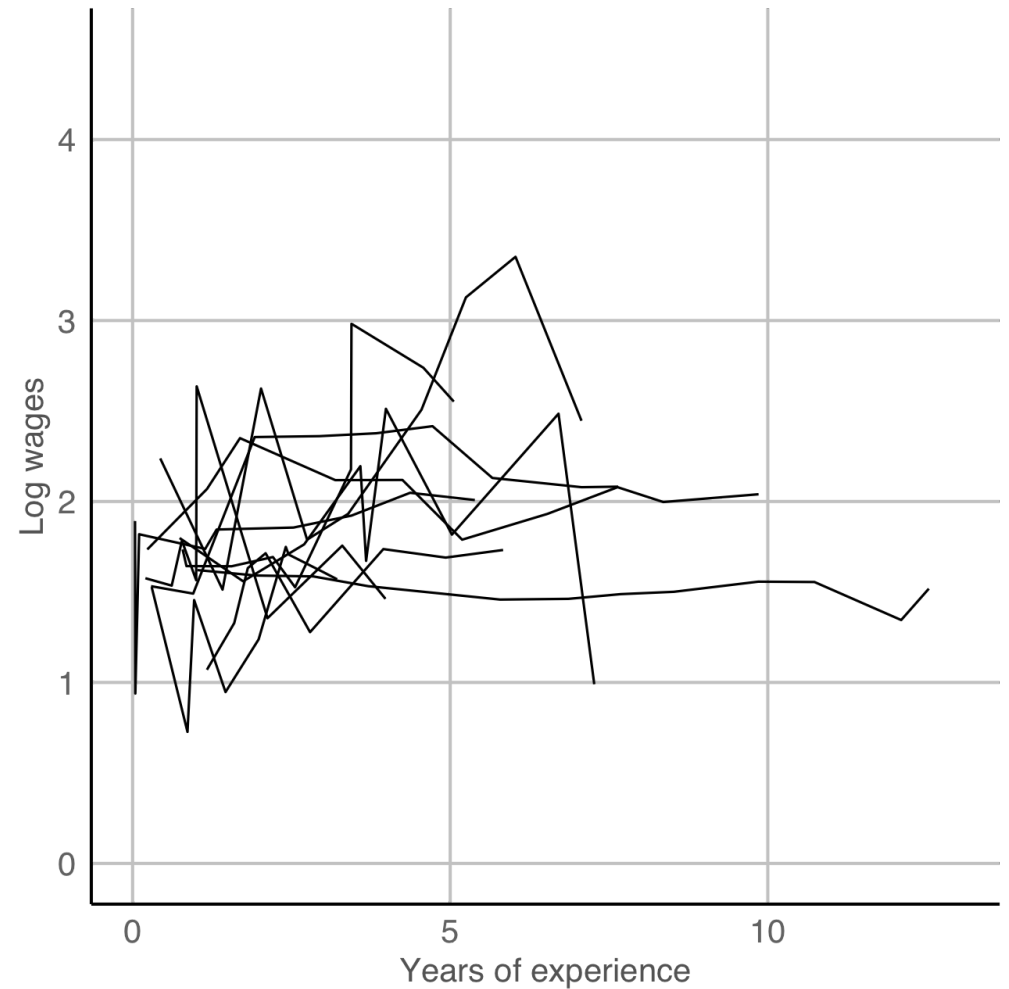
## Case study 4 Wages Part 7/15

Take a spoonful of different lengths

Sample experienced individuals

```
wages %>%  
  add_n_obs() %>%  
  filter(n_obs > 7) %>%  
  sample_n_keys(size = 10) %>%  
  ggplot(aes(x = xp,  
             y = ln_wages,  
             group = id)) +  
  geom_line() +  
  xlim(c(0,13)) + ylim(c(0, 4.5)) +  
  xlab("Years of experience") +  
  ylab("Log wages")
```

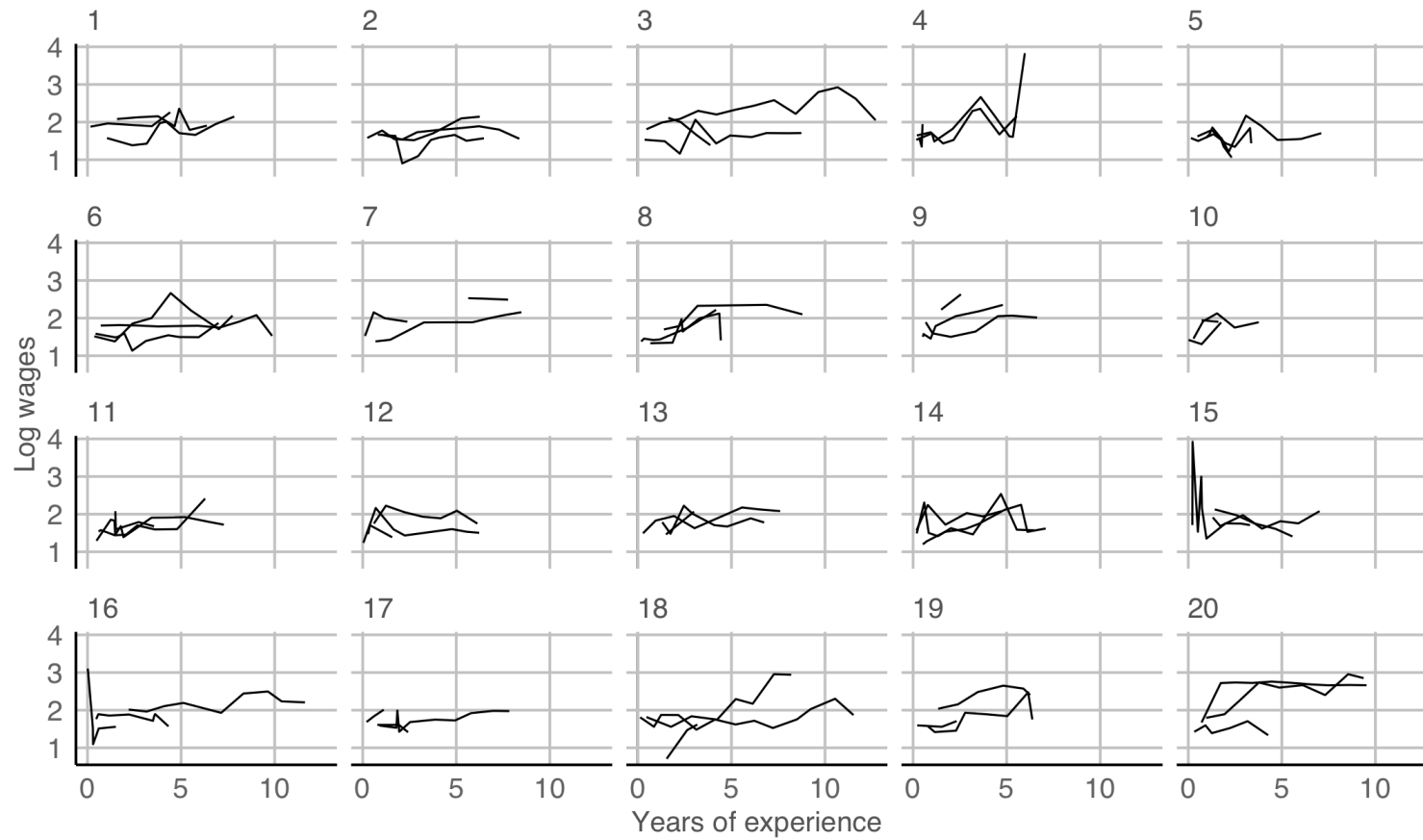
Wages conversion 0.5 = \$1.65; 4.5 = \$90



# Case study **4** Wages Part 8/15



info R





## **Special features**


# Special features

Remember scagnostics?

Compute longnostics for each subject

-  Slope, intercept from simple linear model

-  Variance, standard deviation

-  Jumps, differences

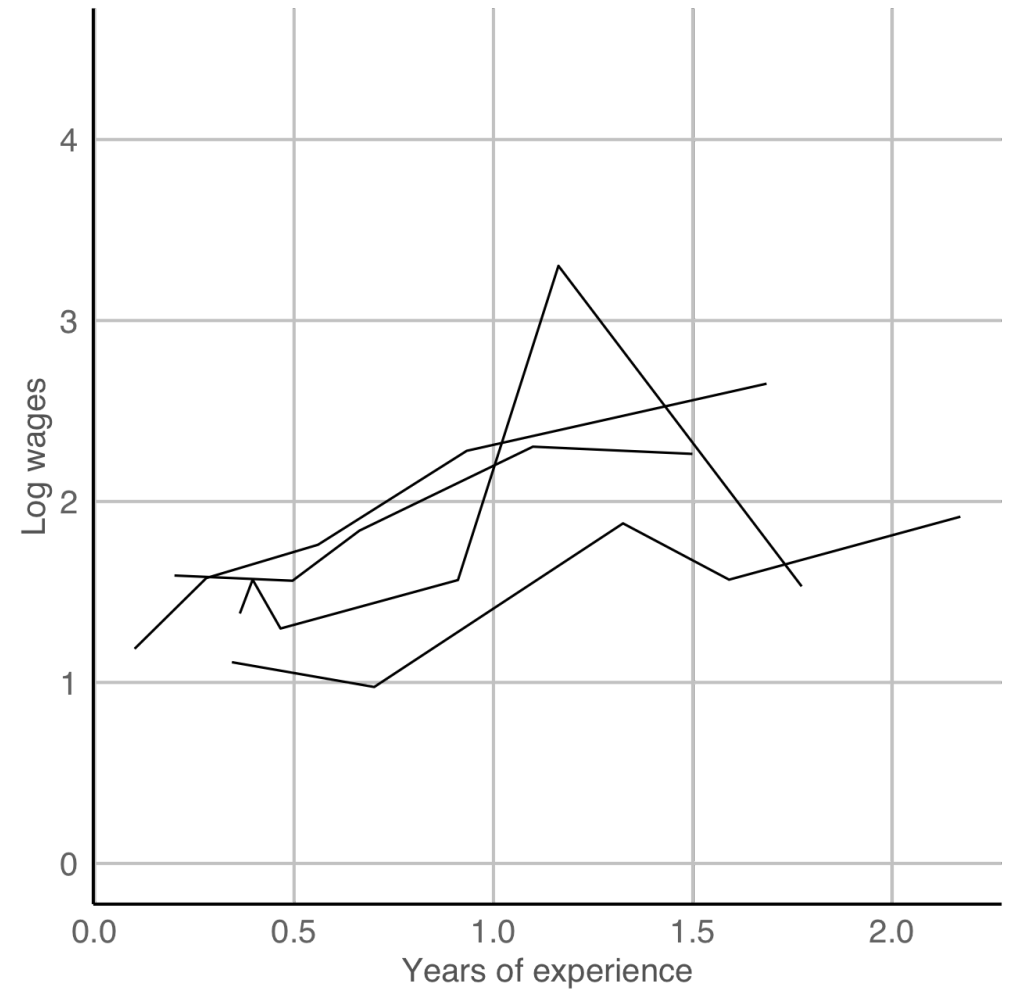
For large collections of time series, take a look at the [feasts](#) package, which has a long list of time series features (tignostics) to calculate.

## Case study 4 Wages Part 9/15

increasing

```
wages_slope <- wages %>%  
  add_n_obs() %>%  
  filter(n_obs > 4) %>%  
  add_key_slope(ln_wages ~ xp) %>%  
  as_tsibble(key = id, index = xp)
```

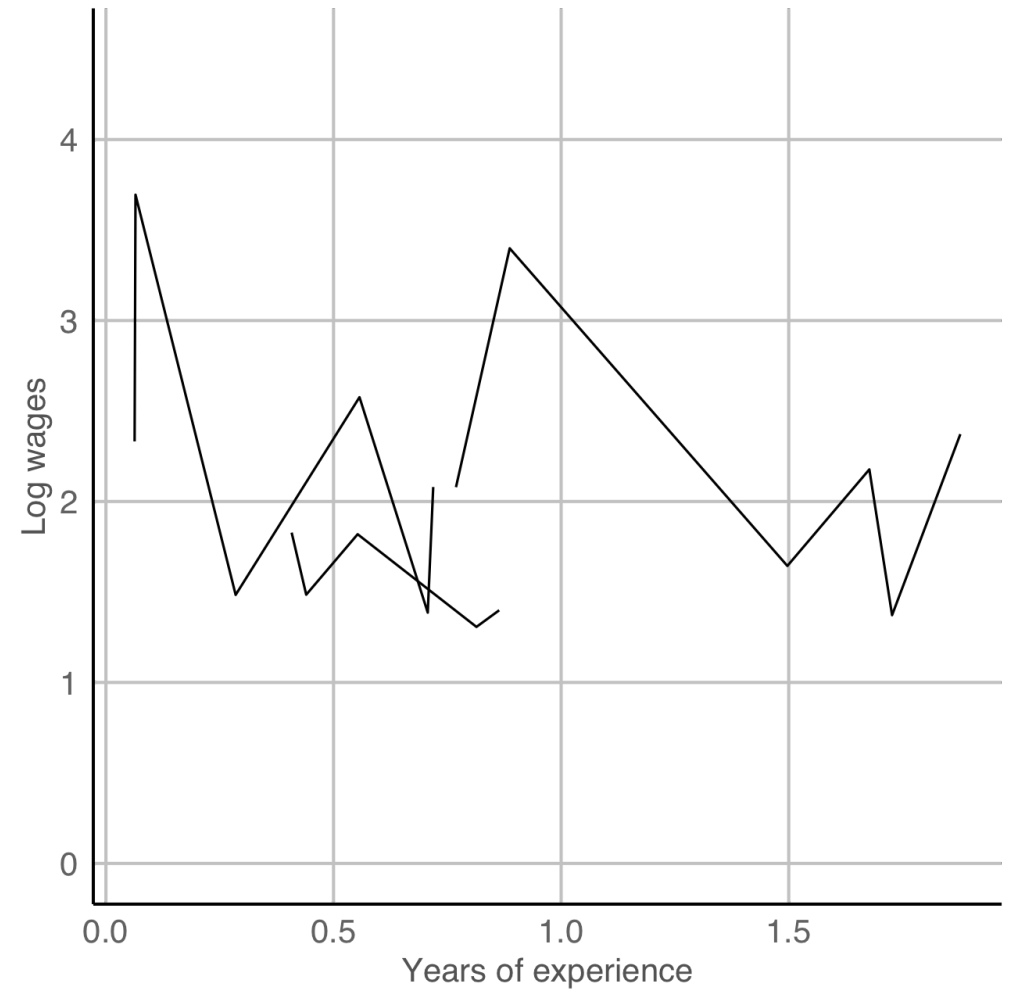
```
wages_slope %>%  
  filter(.slope_xp > 0.4) %>%  
  ggplot(aes(x = xp,  
             y = ln_wages,  
             group = id)) +  
    geom_line() +  
    ylim(c(0, 4.5)) +  
    xlab("Years of experience") +  
    ylab("Log wages")
```



## Case study 4 Wages Part 10/15

decreasing

```
wages_slope %>%  
  filter(.slope_xp < (-0.7)) %>%  
  ggplot(aes(x = xp,  
             y = ln_wages,  
             group = id)) +  
  geom_line() +  
  ylim(c(0, 4.5)) +  
  xlab("Years of experience") +  
  ylab("Log wages")
```



**Longitudinal data needs a special five number summary**

# Summarising individuals

A different style of five number summary

Who is average? Who is different?

Find those individuals who are **representative** of the min, median, maximum, etc of a particular feature, e.g. trend, using `keys_near()`. This reports the individual who is closest to a particular statistic.

`wages_threenum()` returns the three individuals: min, max and closest to the median value.

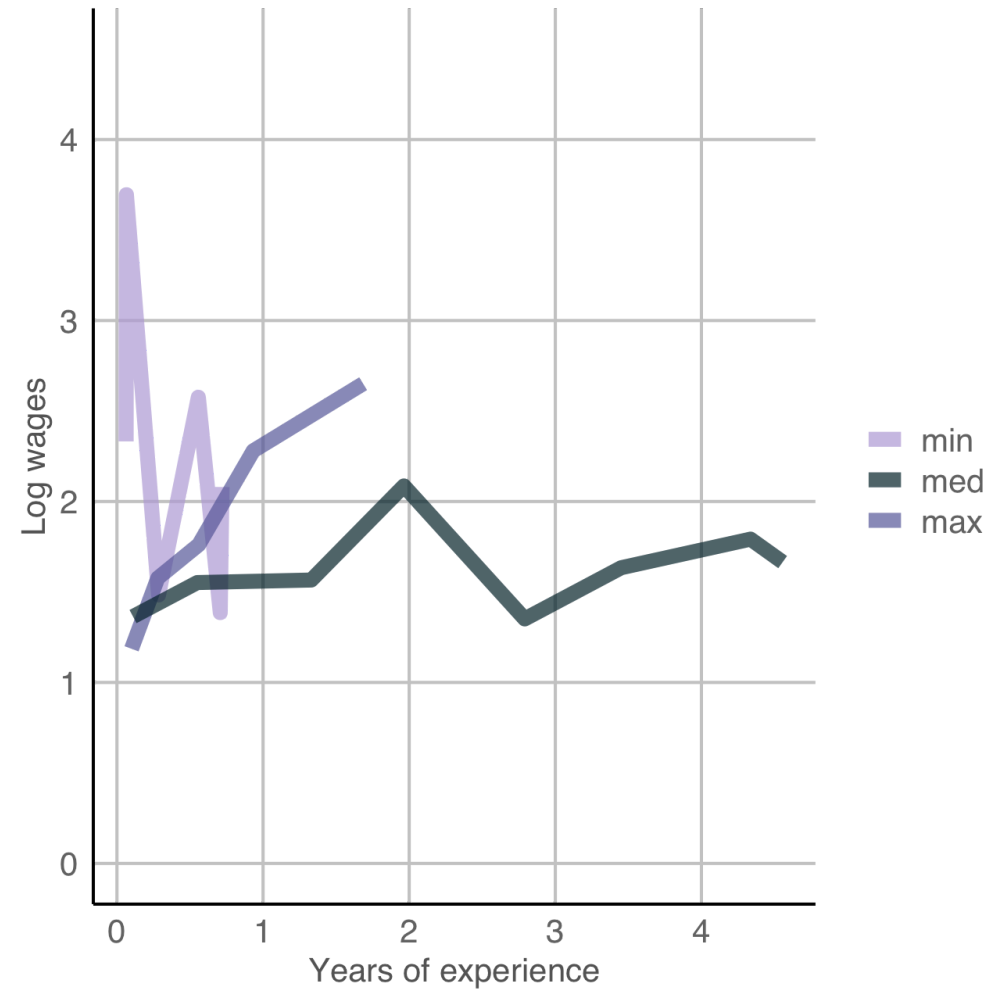
`wages_fivenum()` returns the five individuals: min, max and closest to the median, Q1 and Q3 values.

## Case study 4 Wages Part 11/15

```
wages_threenum <- wages %>%  
  add_n_obs() %>%  
  filter(n_obs > 4) %>%  
  key_slope(ln_wages ~ xp) %>%  
  keys_near(key = id,  
            var = .slope_xp,  
            funs = l_three_num) %>%  
  left_join(wages, by = "id") %>%  
  as_tsibble(key = id, index = xp)
```



Minimum/maximum are short series with substantial decline/incline. Median is very flat, no change in real wages.

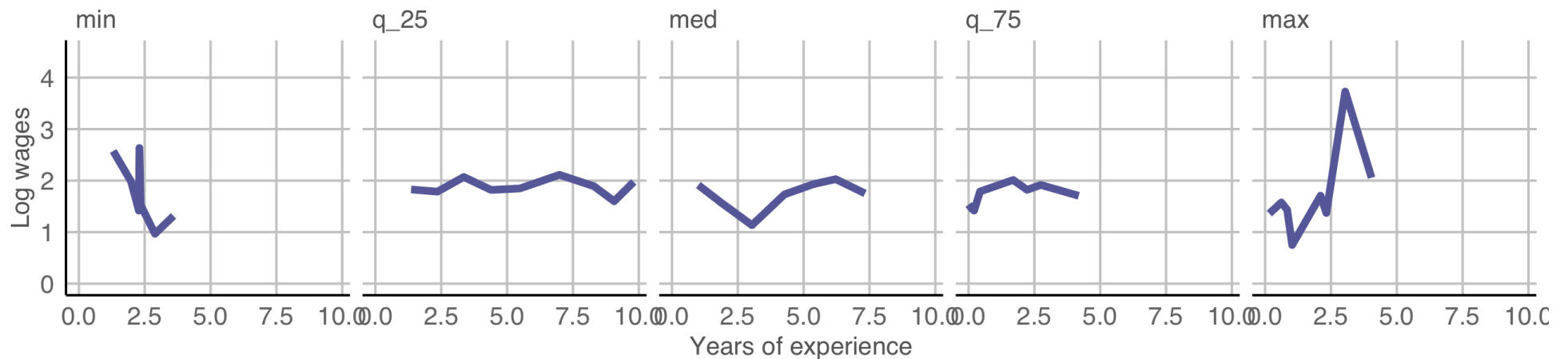


## Case study 4 Wages Part 12/15

```
wages_fivenum <- wages %>%  
  add_n_obs() %>%  
  filter(n_obs > 6) %>%  
  key_slope(ln_wages ~ xp) %>%  
  keys_near(key = id,  
            var = .slope_xp,  
            funs = l_five_num) %>%  
  left_join(wages, by = "id") %>%  
  as_tsibble(key = id, index = xp)
```



Q1 and Q3 are also flat which means that, at least, 50% of the individuals experience no real change in wage.



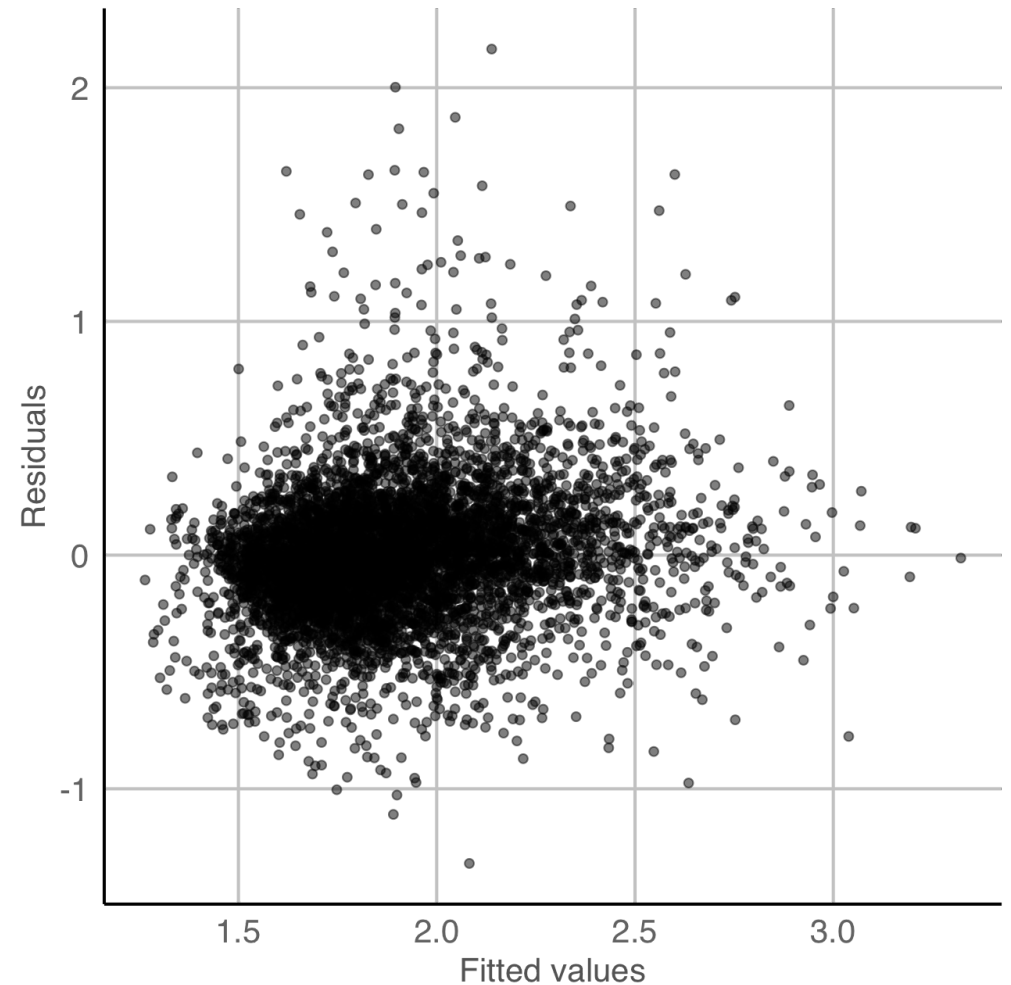


## Case study 4 Wages Part 13/15

### Sculpting spaghetti

Mixed effects model, education as fixed effect, subject random effect using slope.

```
wages_fit_int <-  
  lmer(ln_wages ~ xp + high_grade +  
        (xp |id), data = wages)  
wages_aug <- wages %>%  
  add_predictions(wages_fit_int,  
                  var = "pred_int") %>%  
  add_residuals(wages_fit_int,  
                var = "res_int")
```

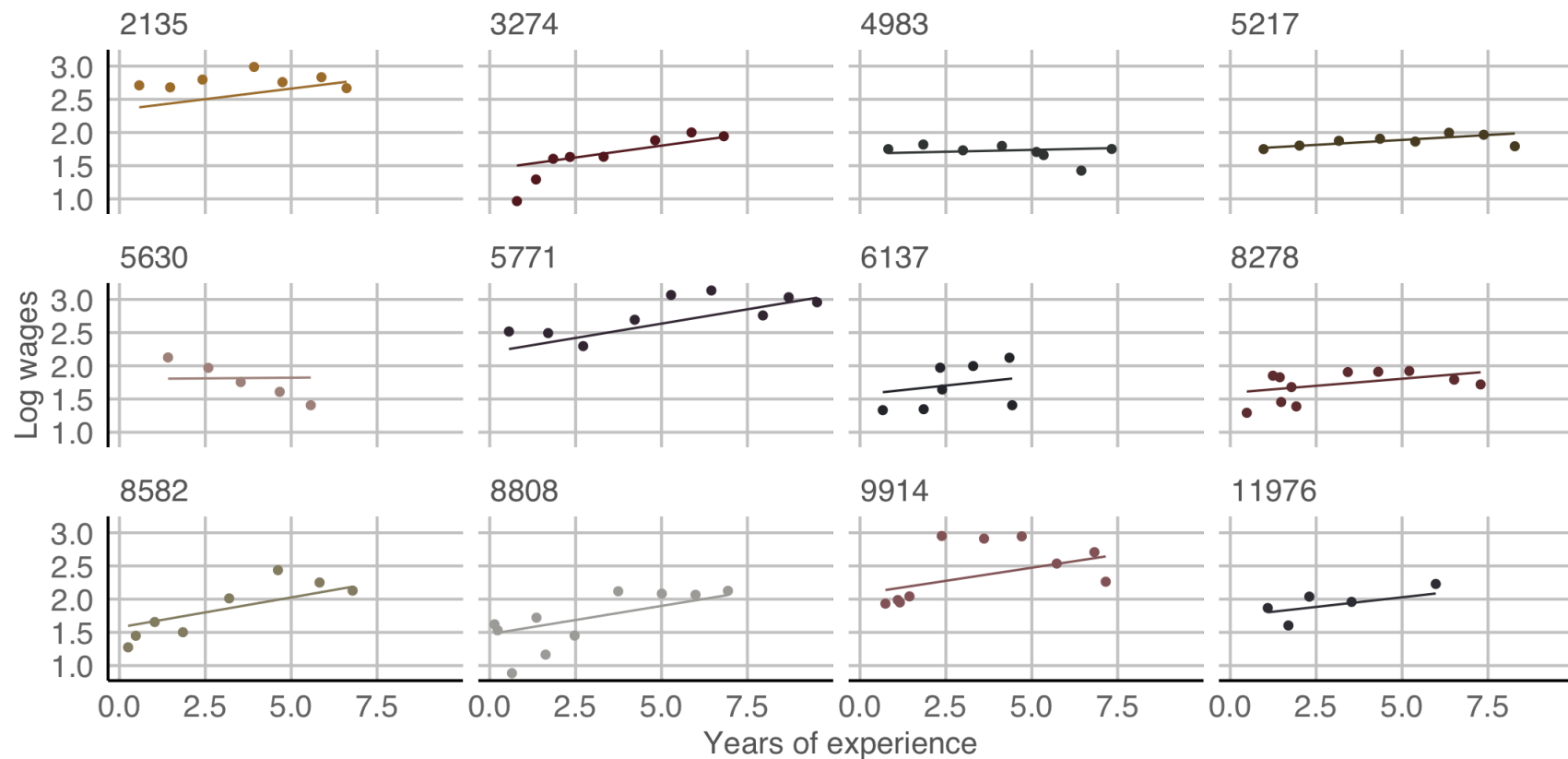


## Case study 4 Wages Part 14/15






R

Model diagnostics: Sample individuals and plot model on the data. Notice individual 5630.



## Case study **4** Wages Part 15/15

### What we learn about wages that we would not have learned without doing EDA

-  The individual wage experience is extremely varied
-  Some individuals see a decline in their wages the longer they are in the workforce
-  Most individuals generally see some (small) increase, on average

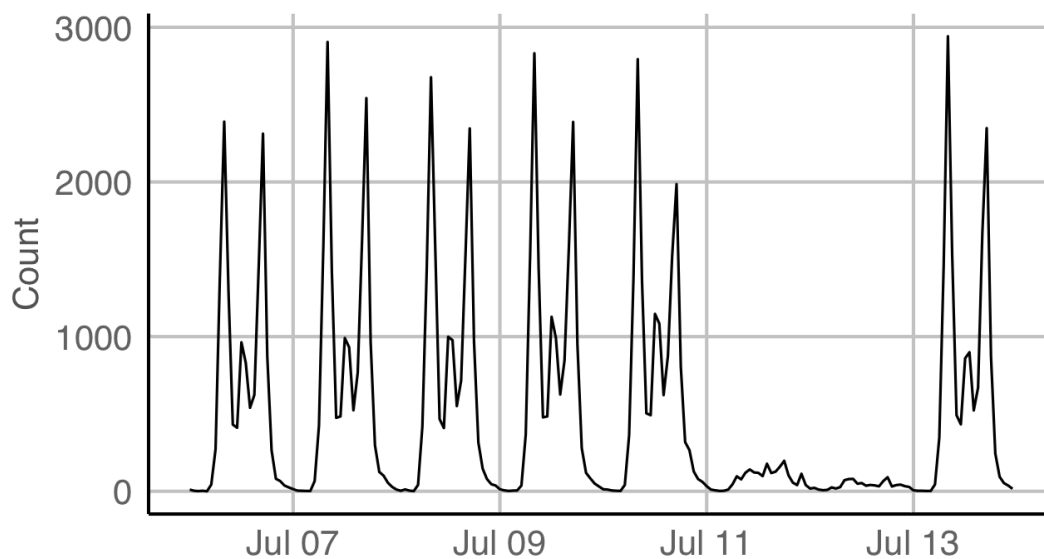


Exploratory analysis of individual temporal patterns can be very really interesting!

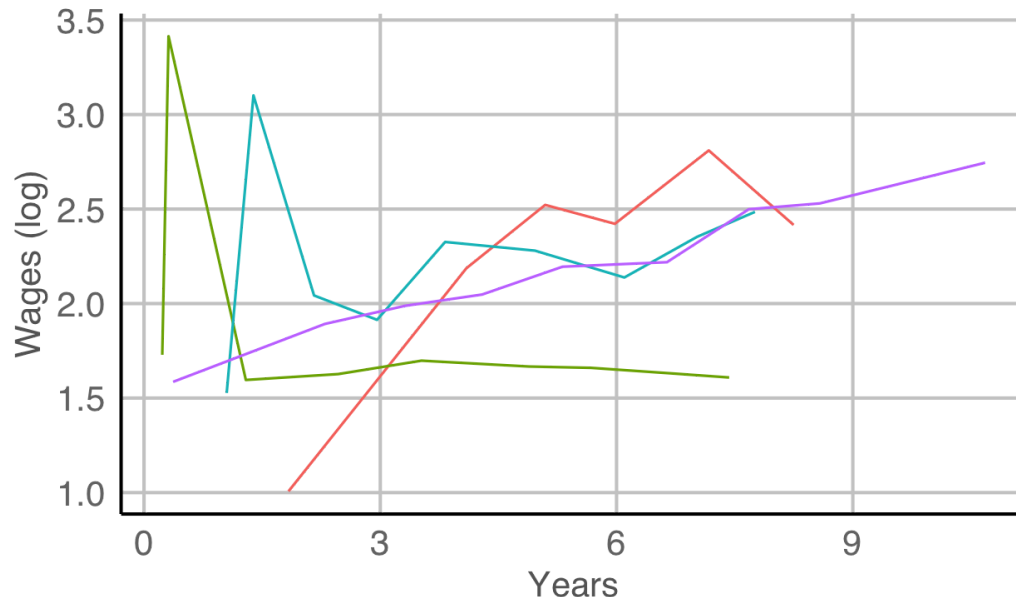


The main difference between **time series** data and **longitudinal** data, is the former is typically regular, complete, may be only one or a few, and the latter is typically of different lengths, different time measurements and a lot.

Time series



Longitudinal



# Resources and Acknowledgement

- 📈 Tidy tools for time series [tidyverts](#)
- 📈 Imputing missings in time using [imputeTS](#)
- 📈 Temporal missings in [tsibble](#)
- 📈 Longitudinal data exploration with [bro1gar](#)
- 📈 Data coding using [tidyverse](#) suite of R packages
- 📈 Slides constructed with [xaringan](#), [remark.js](#), [knitr](#), and [R Markdown](#).



MONASH  
University



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/).

Lecturer: *Di Cook*

✉ [ETC5521.Clayton-x@monash.edu](mailto:ETC5521.Clayton-x@monash.edu)

📅 Week 9 - Session 2

