ETC1010: Introduction to Data Analysis

Week 9, part B

Networks and Graphs

Lecturer: *Nicholas Tierney*

Department of Econometrics and Business Statistics

nicholas.tierney@monash.edu

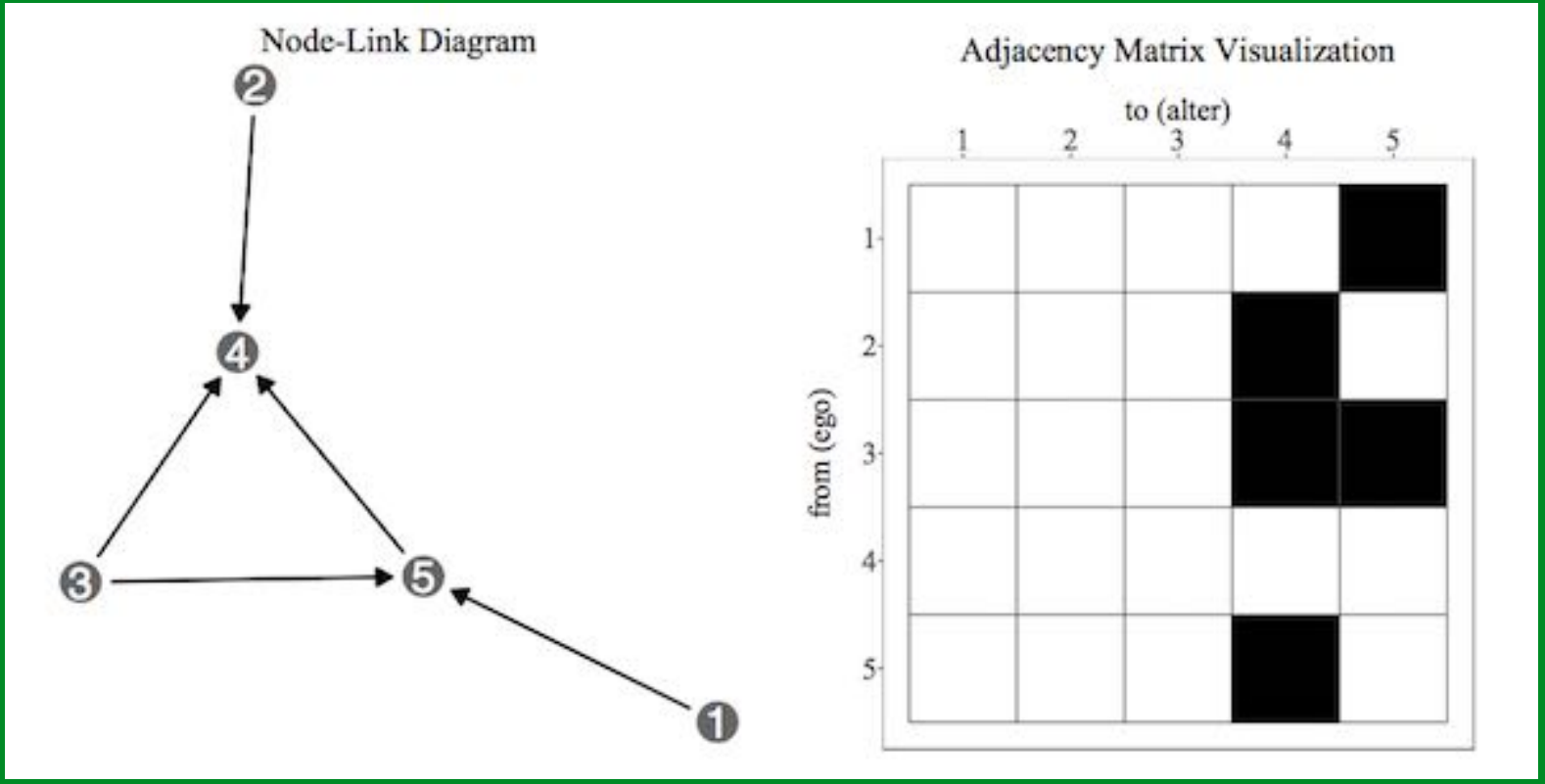
May 2020

Announcements

- Project deadlines:
  - **Deadline 2 (22nd May)** : Team members and team name, data description.
  - **Deadline 3 (29th May)** : Electronic copy of your data, and a page of data description, and cleaning done, or needing to be done.
  - **Deadline 4 (5th June)** : Final version of story board uploaded.
- Practical exam: **DATE from 12pm - 2pm**
- Final Exam: I will provide a review of exam content

Quantitative association matrices

Previous association matrices were black and white:



Quantitative association matrices

- You could have the association between nodes described as real numbers.
- E.g., these are the number of times that these people called each other in the last week:

	Meg	Tay	Yat	Zili	Jess
Meg	0	5	4	1	1
Tay	5	0	4	2	1
Yat	4	4	0	0	0
Zili	1	2	0	0	6
Jess	1	1	0	6	0

Quantitative association matrices

We would need to turn this into an edge data set:

```
## # A tibble: 25 x 3
##    from  to    count
##    <chr> <chr> <dbl>
##  1 Meg   Meg       0
##  2 Tay   Meg       5
##  3 Yat   Meg       4
##  4 Zili  Meg       1
##  5 Jess  Meg       1
##  6 Meg   Tay       5
##  7 Tay   Tay       0
##  8 Yat   Tay       4
##  9 Zili  Tay       2
## 10 Jess  Tay       1
## # … with 15 more rows
```

Quantitative association matrices

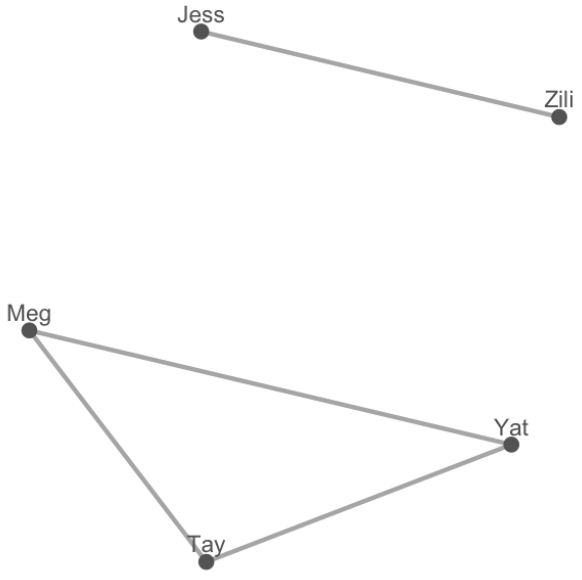
- We need to decide what corresponds to a "connection".
- Let's say they need to have called each other at least 4 times, to be considered connected.

```
d_edges_filter <- d_edges %>% filter(count > 3)

d_edges_filter
## # A tibble: 8 x 3
##   from  to    count
##   <chr> <chr> <dbl>
## 1 Tay   Meg       5
## 2 Yat   Meg       4
## 3 Meg   Tay       5
## 4 Yat   Tay       4
## 5 Meg   Yat       4
## 6 Tay   Yat       4
## 7 Jess  Zili      6
## 8 Zili  Jess      6
```

Association matrices: Make the network diagram.

```
library(geomnet)
set.seed(2020-05-09)
ggplot(data = d_edges_filter,
       aes(
         from_id = from,
         to_id = to)) +
  geom_net(
    layout.alg = "kamadakawai",
    size = 3,
    labelon = TRUE,
    vjust = -0.6,
    ecolour = "grey60",
    directed =FALSE,
    fontsize = 4,
    ealpha = 0.5
    ) +
    theme_net()
```



Data: Last 4 months of currency USD cross-rates in 2018

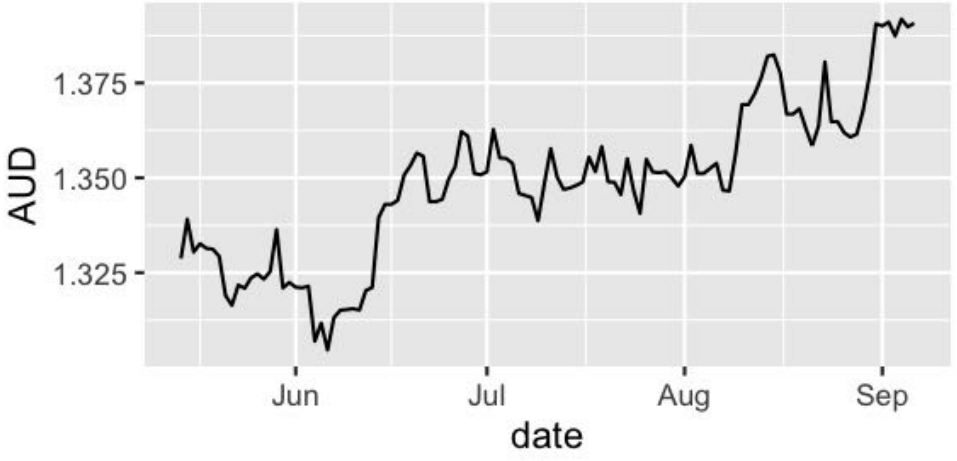
SO let's try this with cross-currency rates across the globe!

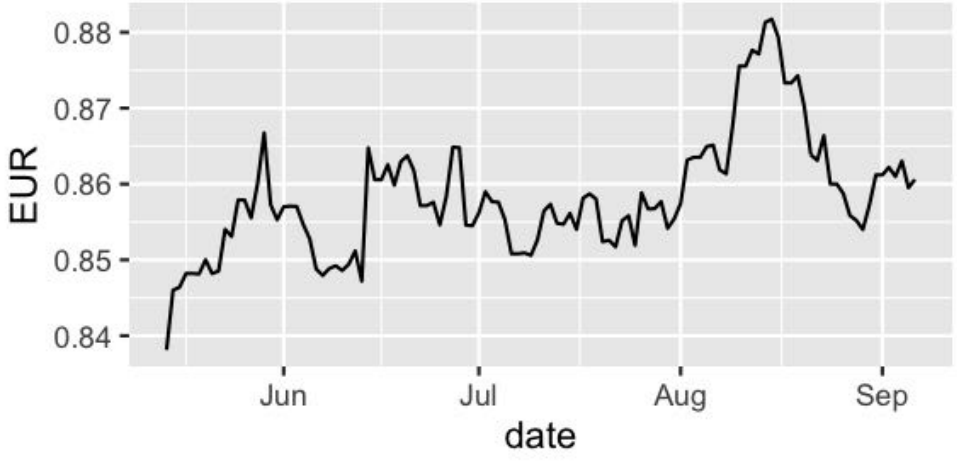
- Data extracted from http://openexchangerates.org/api/historical
- R packages `jsonlite`, processed with `tidyverse`, `lubridate`

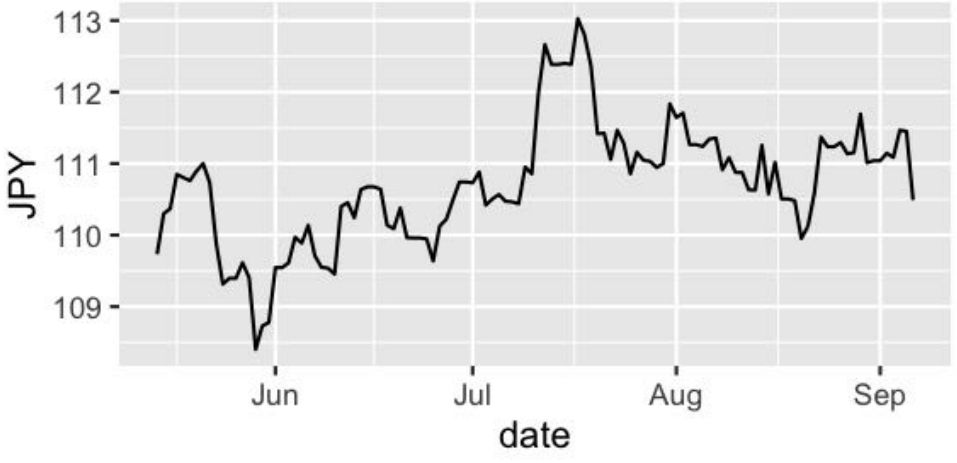
Data: Last 4 months of currency USD cross-rates in 2018

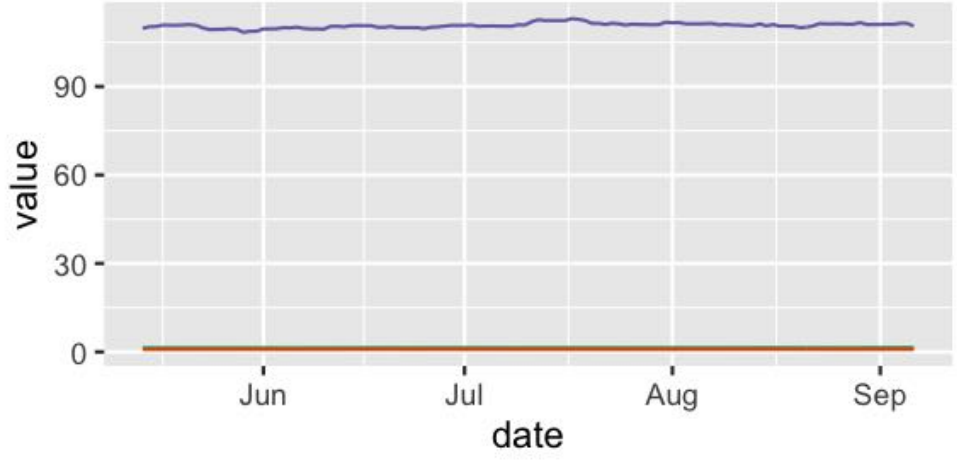
```
## # A tibble: 6 x 171
##   date         AED   AFN   ALL   AMD   ANG   AOA   ARS   AUD   AWG   AZN   BAM   BB
##   <date>     <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl
## 1 2018-05-14  3.67  71.2  106.  485.  1.79  230.  25.0  1.33  1.79  1.70  1.63
## 2 2018-05-15  3.67  71.2  107.  485.  1.80  230.  24.1  1.34  1.79  1.70  1.64
## 3 2018-05-16  3.67  71.0  108.  484.  1.80  232.  24.3  1.33  1.79  1.70  1.66
## 4 2018-05-17  3.67  71.0  108.  483.  1.80  233.  24.3  1.33  1.79  1.70  1.66
## 5 2018-05-18  3.67  71.0  108.  483.  1.80  233.  24.4  1.33  1.79  1.70  1.66
## 6 2018-05-19  3.67  70.9  108.  482.  1.79  233.  24.4  1.33  1.79  1.70  1.66
## # … with 158 more variables: BDT <dbl>, BGN <dbl>, BHD <dbl>, BIF <dbl>, BMD <dbl>,
## #   BND <dbl>, BOB <dbl>, BRL <dbl>, BSD <dbl>, BTC <dbl>, BTN <dbl>, BWP <dbl>,
## #   BYN <dbl>, BZD <dbl>, CAD <dbl>, CDF <dbl>, CHF <dbl>, CLF <dbl>, CLP <dbl>,
## #   CNH <dbl>, CNY <dbl>, COP <dbl>, CRC <dbl>, CUC <dbl>, CUP <dbl>, CVE <dbl>,
## #   CZK <dbl>, DJF <dbl>, DKK <dbl>, DOP <dbl>, DZD <dbl>, EGP <dbl>, ERN <dbl>,
## #   ETB <dbl>, EUR <dbl>, FJD <dbl>, FKP <dbl>, GBP <dbl>, GEL <dbl>, GGP <dbl>,
## #   GHS <dbl>, GIP <dbl>, GMD <dbl>, GNF <dbl>, GTQ <dbl>, GYD <dbl>, HKD <dbl>,
## #   HNL <dbl>, HRK <dbl>, HTG <dbl>, HUF <dbl>, IDR <dbl>, ILS <dbl>, IMP <dbl>,
## #   INR <dbl>, IQD <dbl>, IRR <dbl>, ISK <dbl>, JEP <dbl>, JMD <dbl>, JOD <dbl>,
## #   JPY <dbl>, KES <dbl>, KGS <dbl>, KHR <dbl>, KMF <dbl>, KPW <dbl>, KRW <dbl>,
## #   KWD <dbl>, KYD <dbl>, KZT <dbl>, LAK <dbl>, LBP <dbl>, LKR <dbl>, LRD <dbl>,
```

Data: Last 4 months of currency USD cross-rates in 2018





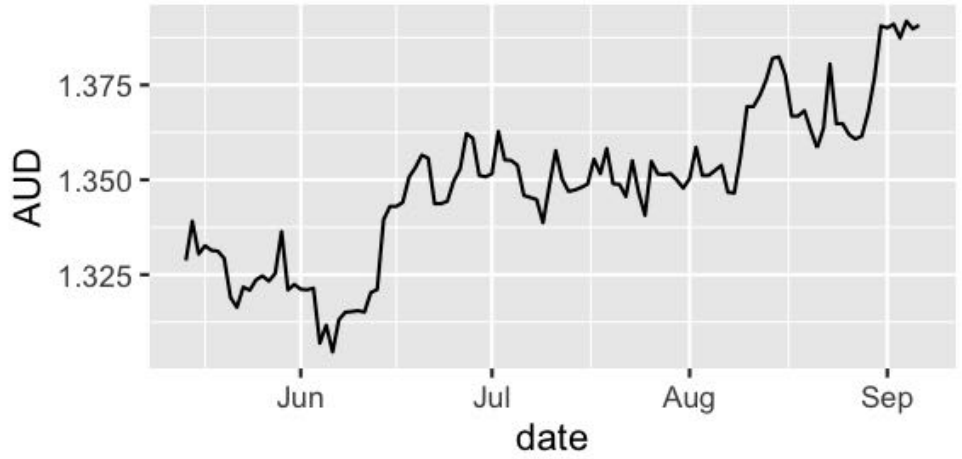


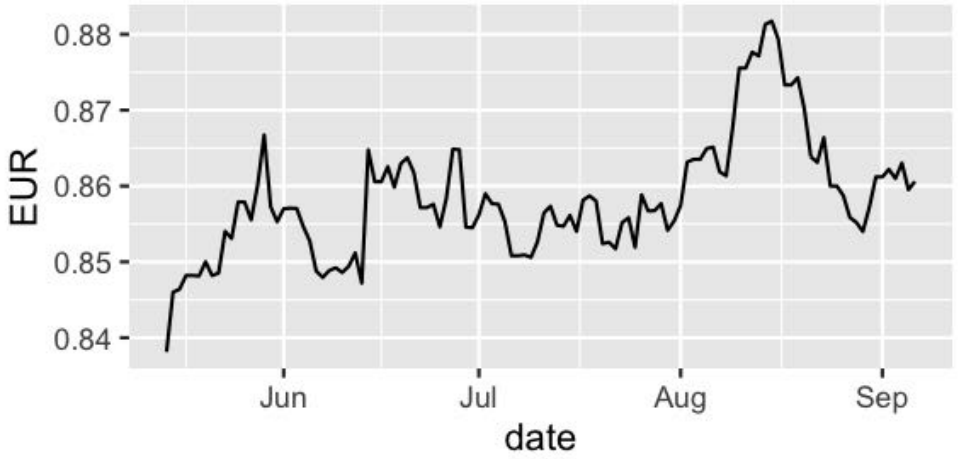


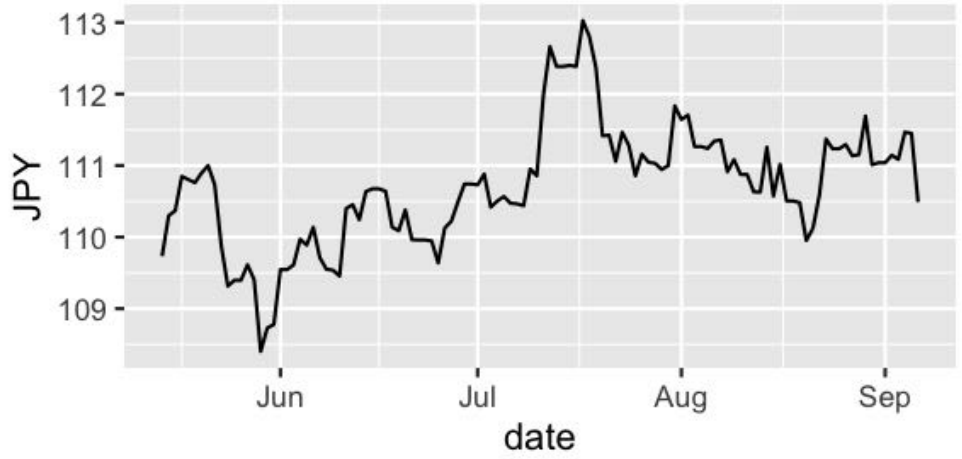
Your turn: Rstudio

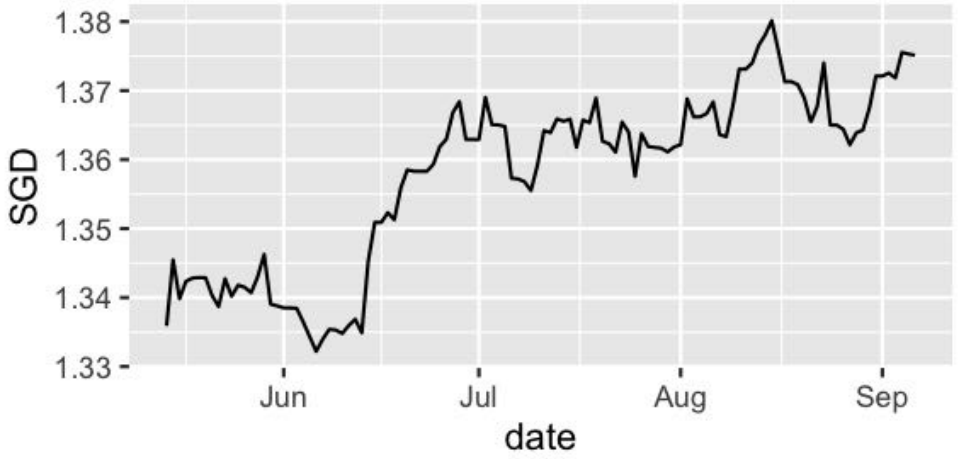
Make some plots (or google) to answer these questions

- Is the NZD more similar to AUD, EUR, or JPY? (What currency is NZD?)
- Is SGD more similar to AUD, EUR, or JPY? (What currency is SGD?)
- How many currencies are there in the British Isles?


AUD
1.375
1.350
1.325
Jun
Jul
Aug
Sep
date


EUR
0.88
0.87
0.86
0.85
0.84
Jun
Jul
Aug
Sep
date


JPY
113
112
111
110
109
Jun
Jul
Aug
Sep
date


SGD
1.38
1.37
1.36
1.35
1.34
1.33
Jun
Jul
Aug
Sep
date

Pre-processing: Keep currencies that change

- Some currencies don't change very much.
- These should be filtered from the analysis, because in a study of currency movement, if it doesn't move then there is nothing more to be said.

Pre-processing: Keep currencies that change

- To filter out these currencies we use a statistic called coefficient of variation:

$$CoefVariation = \frac{\sigma}{\mu}$$

- Measures standard deviation of currency relative to the mean.
- For high means, we expect a currency to change more.
- That is, relatively the standard deviation would be larger to consider it to be changing.

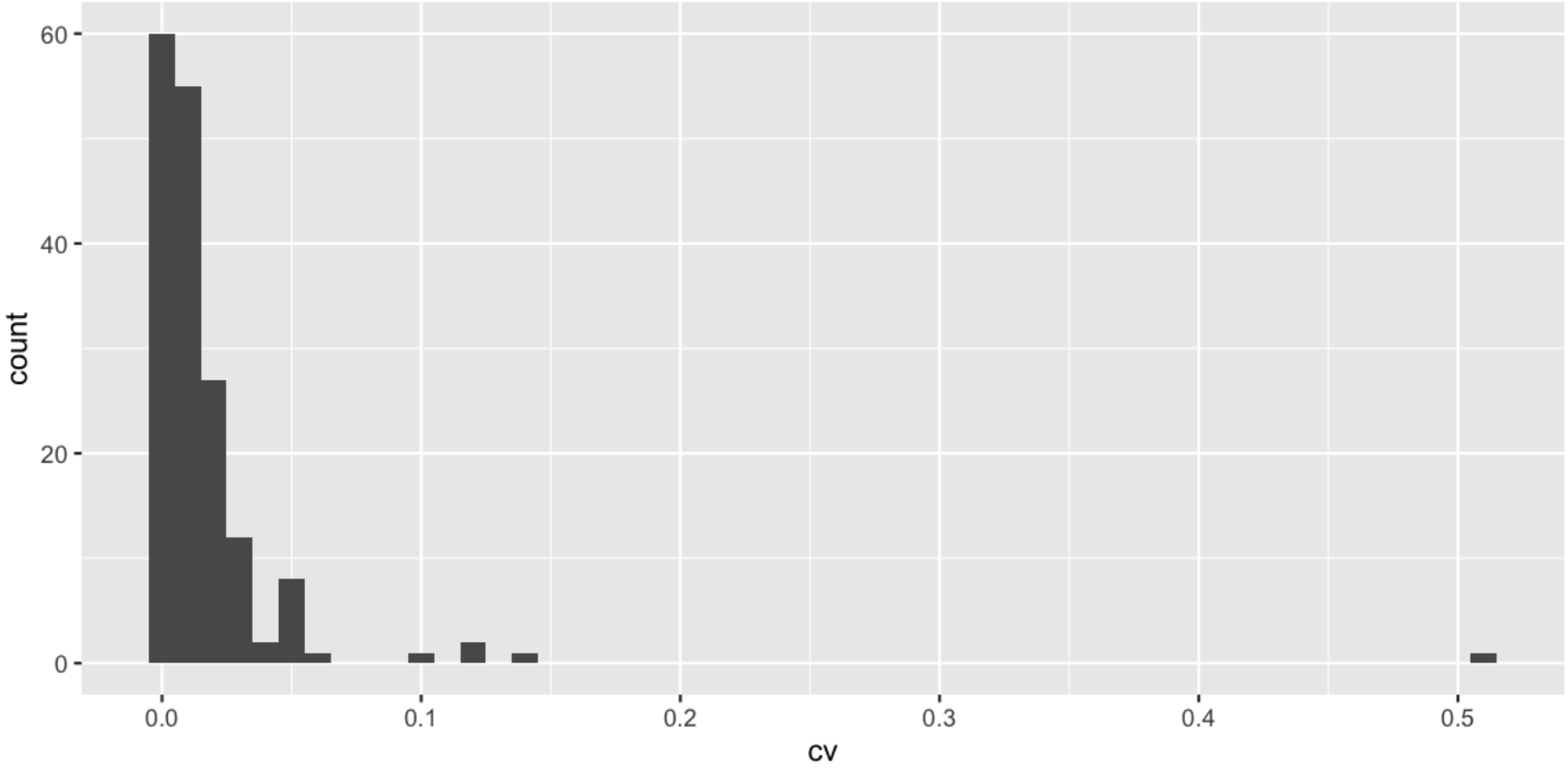
Computing CV

Strategy pivot to long form then group and summarize currency values

```
# Compute coefficient of variation. We will only analyse
# currencies that have changes substantially over this time.
cv <- function(x){
  sd(x)/mean(x)
}

rates_cv <- rates %>%
  pivot_longer(cols = -date, names_to = "currency") %>%
  group_by(currency) %>%
  summarise(cv = cv(value))
```

Distrubtion of CV values



Identify currencies with CVs below the first quantile

```
rates_stable <- rates_cv %>%
  filter(cv < quantile(cv, 0.25))
```

Filter out low cv currencies using pivot and an anti join

```
rates_sub <- rates %>%
  pivot_longer(cols = -date, names_to = "currency") %>%
  anti_join(rates_stable)
rates_sub
## # A tibble: 14,732 x 3
##    date       currency   value
##    <date>     <chr>      <dbl>
##  1 2018-05-14 AFN         71.2
##  2 2018-05-14 ALL        106.
##  3 2018-05-14 ANG          1.79
##  4 2018-05-14 AOA        230.
##  5 2018-05-14 ARS         25.0
##  6 2018-05-14 AUD          1.33
##  7 2018-05-14 BAM          1.63
##  8 2018-05-14 BDT         84.7
##  9 2018-05-14 BGN          1.64
## 10 2018-05-14 BIF       1767.
## # … with 14,722 more rows
```

Remove currencies that are not currencies

Some of the currencies ... aren't really currencies. Google these ones: XAG, XDR, XPT - what are they?

Remove currencies that are not currencies

```
# Remove non-currencies
rates_dropped <- rates_sub %>%
  filter(!currency %in% c("ALL", "XAG", "XDR", "XPT"))
```

XAG is Gold XPT is Platinum XDR is special drawing rights

Standardize the currencies

To examine overall trend regardless of actual USD cross rate, standardise the values to have mean 0 and standard deviation 1.

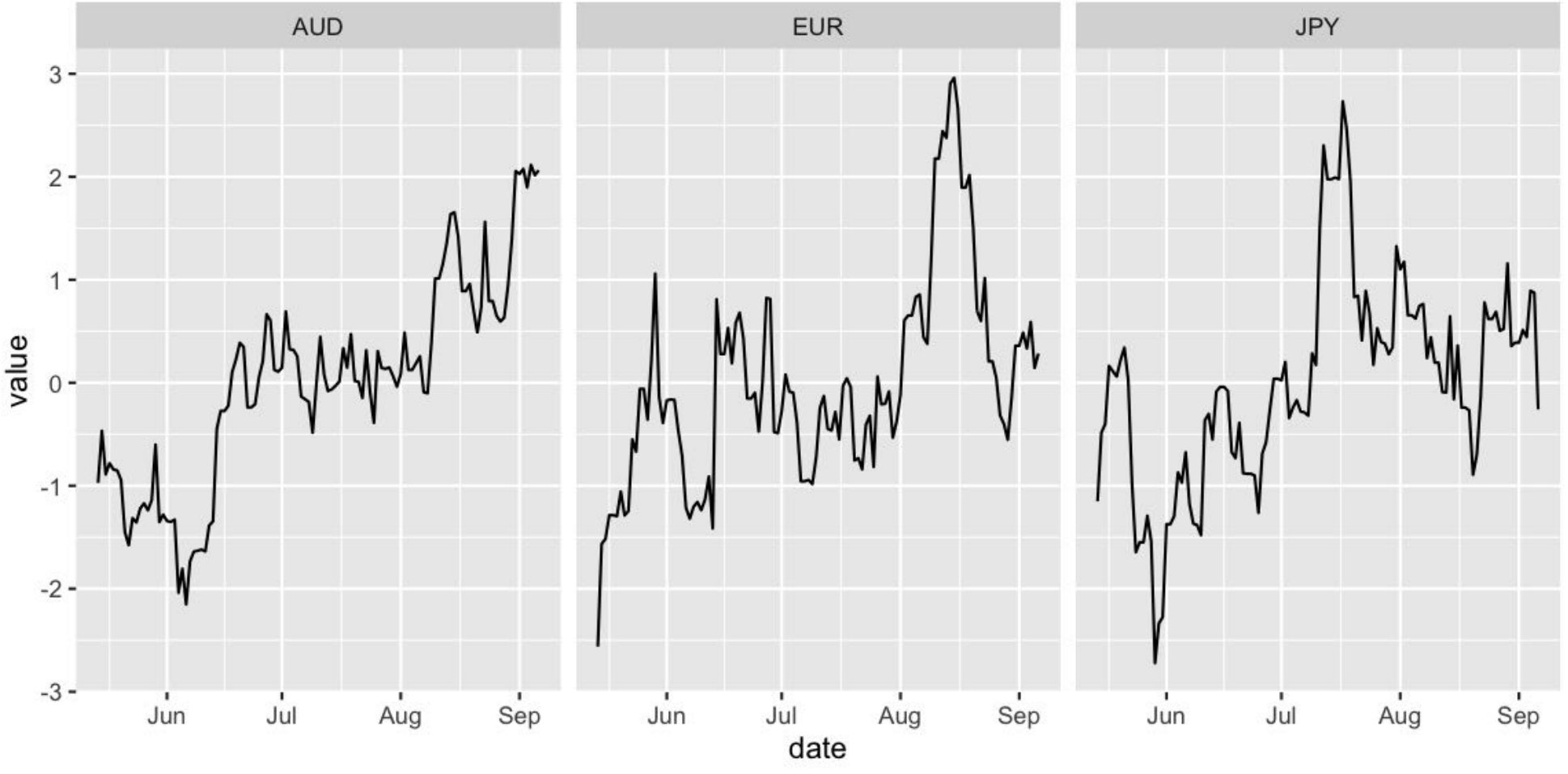
```
scale01 <- function(x) (x - mean(x)) / sd(x)
```

Rescale all values to have standardised values

Use `group_by()` plus `mutate()`!

```
rates_scaled <- rates_dropped %>%
  group_by(currency) %>%
  mutate(value = scale01(value))
```

Standardize the currencies



Compute distances between all pairs of currencies

Euclidean distance is used to compute similarity between all pairs of currencies.

$$d_{ij} = \sqrt{\sum_{i=1}^{t} (C_{1i} - C_{2i})^2}$$

Compute distances between all pairs of currencies

We need to put our data back in wide form! And then turn it into a matrix.

```
rates_wide <- rates_scaled %>%
    pivot_wider(id_cols = "date", names_from = "currency") %>%
    select(-date)

# compute distance between currencies, rows <--> columns
rates_wide_t <- t(rates_wide)
```

Use built in function to compute distance

```
currency_dist <- as.matrix(dist(rates_wide_t,
                                diag = TRUE,
                                upper = TRUE))

currency_dist[1:5, 1:5]
##           AFN      ANG      AOA      ARS      AUD
## AFN 0.000000 8.044527 7.315939 8.014165 7.970993
## ANG 8.044527 0.000000 5.628321 9.601101 7.277124
## AOA 7.315939 5.628321 0.000000 5.760894 5.299254
## ARS 8.014165 9.601101 5.760894 0.000000 5.983452
## AUD 7.970993 7.277124 5.299254 5.983452 0.000000
```

A note on distance matrices:

- A distance matrix is the inverse of an association matrix.
- A distance matrix close to 0 means the pair are most similar.
- For an association matrix far from zero means the pair are close.
- Either can be used to generate a network.

Create network: Pivot data into long form, filter based on similarity

Here only the pairs of currencies who are closer than "4" to each other are kept.

```
distance_tbl <-  currency_dist %>%
  as.data.frame() %>%
  rownames_to_column(var = "from_currency") %>%
  pivot_longer(-from_currency,
               names_to = "to_currency",
               values_to = "distance") %>%
  filter(distance < 4 ) %>%
  filter(from_currency != to_currency)
```

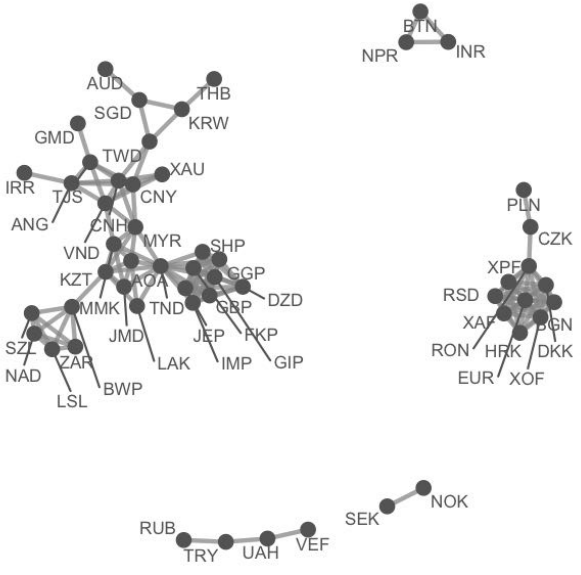
Create network: Gather data into long form, filter based on similarity

Here only the pairs of currencies who are closer than "4" to each other are kept.

```
distance_tbl
## # A tibble: 266 x 3
##    from_currency to_currency distance
##    <chr>         <chr>           <dbl>
##  1 ANG           CNH              2.98
##  2 ANG           CNY              3.24
##  3 ANG           IRR              3.73
##  4 ANG           TJS              3.60
##  5 ANG           VND              3.42
##  6 AOA           JMD              3.66
##  7 AOA           KZT              2.11
##  8 AOA           LAK              3.55
##  9 AOA           MMK              2.19
## 10 AOA           MYR              2.17
## # … with 256 more rows
```

Network laid out

```
set.seed(10052016)
ggplot(data = distance_tbl,
       aes(
         from_id = from_currency
         to_id = to_currency
         )) +
  geom_net(
    size = 3,
    labelon = TRUE,
    repel = TRUE,
    ecolour = "grey60",
    fontsize = 3,
    ealpha = 0.5
    ) +
    theme_net() +
    theme(
      legend.position = "bottom"
      )
```



Your turn

- Make a plot of the AUD vs the SGD (using the standardised units). Do they look like they are trending together as suggested by the network?
- Try out the remaining lab exercises

Flexdashboard

[demo]

Flexdasboard

Here is a list, in order of viewing.

1. Sharon Machlis: R language tip: Easy dashboards with flexdashboard https://www.youtube.com/watch?v=_oDfBVr9wmQ
2. Jonathan Ng's series:
   - 5 Minute Dashboard with R Shiny Flex Dashboards https://www.youtube.com/watch?v=45h71BFbL1w: Getting set up with shiny, to have inputs and reactive plots. Uses an igraph example.
   - Flexdashboard Cheat Sheet https://www.youtube.com/watch?v=gkQvhMA24ig: Layout explanations. Nice style of making changes and exploring the result
   - Dyanmic Dashboard Filters with R, Shiny Flex Dashboards https://www.youtube.com/watch?v=MBNdyRQIvE4: Reasonable getting started with shiny elements.

Flexdashboard

1. Jonathan Ng's series (continued):

- Build a Dashboard in 10 Seconds with R Shiny Flexdashboard https://www.youtube.com/watch?v=6WTaGEOVJ6s: Advanced R coding. Starts from a sample flexdashboard with inputs and reactives, and adds more advanced elements to it. (Follows Dyanmic Dashboard Filters with R, Shiny Flex Dashboards)
- Load R Shiny Flexdashboards Faster https://www.youtube.com/watch?v=MlfHf8PpX5E&

A note on presenting your project

- We suggest making recording a group presentation with zoom, and uploading to youtube as an unlisted video
- Time limit of 5 minutes
- You can use basic software like Quicktime to trim the starts and ends of the videos
- I will post more details on how to post videos onto youtube soon.