Appendix to 'cubble: An R Package for Organizing and Wrangling Multivariate Spatio-temporal Data'

Sherry Zhang, Dianne Cook, Ursula Laa, Nicolas Langrené, Patricia Menéndez

2022-06-14

This is the supplementary material for the main paper, containing an extended example following example 5.2 to highlight how **cubble** can be used to deal with data that has a hierarchical structure. It also describes in detailed the process to create linking plots. Furthermore, this appendix contains additional information about the sources of the data sets use in the paper and the necessary code for extracting and preparing the data as used in the main text with the goal to ensure reproducibility.

1 Extension of example 5.2: Australian precipitation pattern in 2020

In the previous example, some overlapping of the glyphs occurred for a few nearby stations such as the pairs (151E, 34S) and (152E, 33S). This is a problem when mapping more stations at the national level. Aggregation can be helpful in grouping series into clusters before visualising the clusters with a glyph map. This new example shows how to organise data at both levels with `switch_key()`.

The data `climate_full`, also extracted from the GHCN, records daily precipitation and maximum/minimum temperature for 640 stations in Australia from 2016 to 2020. A simple k-means algorithm based on the distance matrix between stations is used to create 20 clusters. The data `station_nested` is a nested cubble with a cluster column indicating the group to which each station belongs. More advanced clustering algorithms can be used as well, as long as they provide a mapping from each station to a cluster.

```
station_nested <- climate_full %>% mutate(cluster = ...)
```

To create a group-level cubble, use `switch_key()` with the new key variable, `cluster`:

```
cluster_nested <- station_nested %>% switch_key(cluster)
```

With the group-level cubble, `get_centroid()` is useful to compute the centroid of each cluster, which will be used as the major axis for the glyph map later:

```
cluster_nested <- cluster_nested %>% get_centroid()
```

Long form cubble at both levels can be accessed through stretching the nested form. With access to both station and cluster-level cubbles, various plots can be made to understand the cluster. Figure **??** shows two example plots that can be made with this data. Subplot A is a glyph map made with the cluster level cubble in the long form and subplot B inspects the station membership of each cluster using the station level cubble in the nested form.

```
coords <- cbind(prcp_aus$long, prcp_aus$lat)
dist_raw <- geosphere::distm(coords, coords)

station_long <- cubble::prcp_aus %>%
  face_temporal(ts) %>%
  mutate(wk = lubridate::week(date)) %>%
  group_by(wk) %>%
  summarise(prcp = sum(prcp, na.rm = TRUE)) %>%
  unfold(cluster)

set.seed(123)
station_nested <- station_long %>%
  face_spatial() %>%
  strip_rowwise() %>%
  mutate(cluster = kmeans(dist_raw,centers = 20, nstart = 500)$cluster)
save(station_nested, file = here::here("data/station_nested.rda"))
```

```
load(here::here("data/station_nested.rda"))
cluster_nested <- station_nested %>%
  switch_key(cluster) %>%
  get_centroid()

cluster_long <- cluster_nested %>%
  face_temporal() %>%
  group_by(wk) %>%
  summarise(prcp = mean(prcp, na.rm = TRUE)) %>%
  unfold(cent_long, cent_lat)
```

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

```
state_map <- rmapshaper::ms_simplify(ozmaps::abs_ste, keep = 2e-3)
```

```
## Registered S3 method overwritten by 'geojsonlint':
##   method         from
##   print.location dplyr
```

```
ggplot_smooth <- cluster_long %>%
  ggplot() +
  geom_smooth(aes(x = wk, y = prcp, group = cluster), span = 0.4)

smoother <- layer_data(ggplot_smooth) %>%
  left_join(cluster_long %>% select(cluster, cent_long, cent_lat), by = c("group" = "cluster"))
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
## Adding missing grouping variables: `id`
```

```
p1 <- ggplot(data = smoother,
             aes(x_minor = x, y_minor = y, x_major = cent_long, y_major = cent_lat)) +
  geom_sf(data = state_map, inherit.aes = FALSE,
          color = "grey80", alpha = 0.4, linetype = 3) +
```

```
  geom_text(data = cluster_nested,
            aes(x = cent_long, y = cent_lat, label = cluster), inherit.aes = FALSE) +
  geom_glyph(height = 2, width = 4) +
  theme_void()

p2 <- ggplot() +
  geom_sf(data = state_map, inherit.aes = FALSE,
          color = "grey80", alpha = 0.4, linetype = 3) +
  geom_point(data = station_nested, aes(x = long, y = lat), size = 0.5) +
  ggforce::geom_mark_hull(data = cluster_nested %>% tidyr::unnest(hull),
                          expand = 0, radius = 0,
                          aes(x = long, y = lat, group = cluster)) +
  theme_void()
```

```
## Registered S3 method overwritten by 'ggforce':
##   method           from
##   scale_type.units units
```

```
## Warning: The concaveman package is required for geom_mark_hull
```

```
(p1 | p2) + plot_annotation(tag_levels = 'A')
```

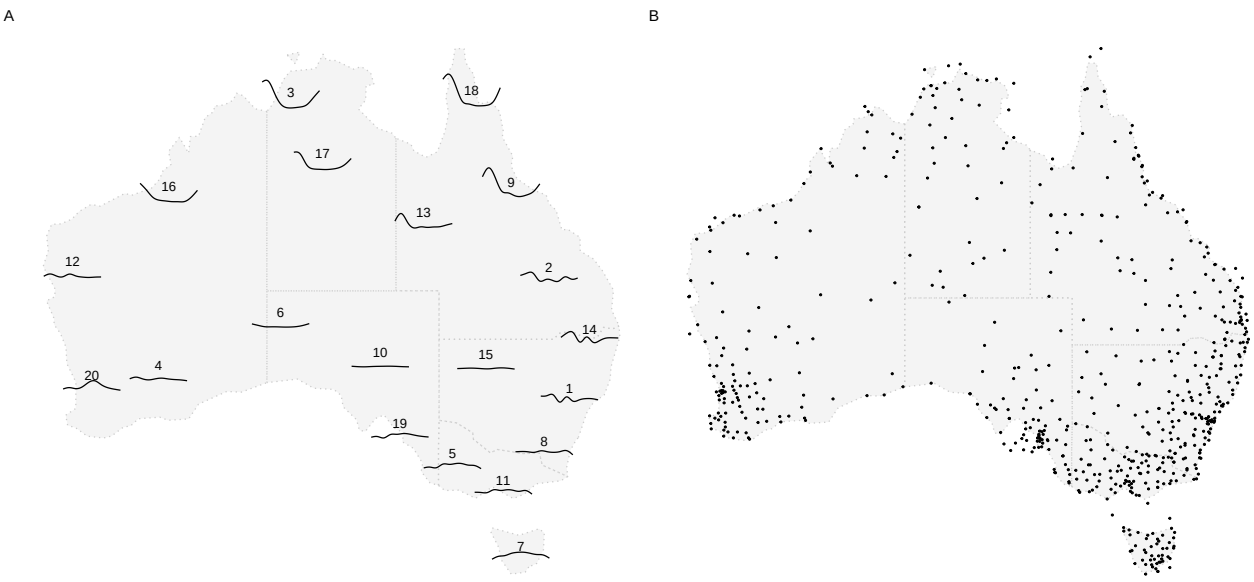

Figure 1: Profile of aggregated precipitation at 639 weather stations in Australia. Subplot A shows the glyph map of the weekly averaged precipitation of each cluster. The group number is printed in the middle of the y minor axis and can be used as a reference line to read the magnitude. Subplot B shows the station membership of each cluster.

2 Additional illustration on multiple linked plots

This figure is a supplement to Section 4.3 of the main paper, illustrating how linking from the time series plot to the map is achieved.

```
## Warning in knitr::include_graphics(here::here("figures/diagram-keynotes/
## diagram-keynotes.005.png")): It is highly recommended to use relative paths for
## images. You had absolute paths: "/Users/pmen0008/Google Drive (2)/ Private/PhD
## supervision/paper-cubble/figures/diagram-keynotes/diagram-keynotes.005.png"
```

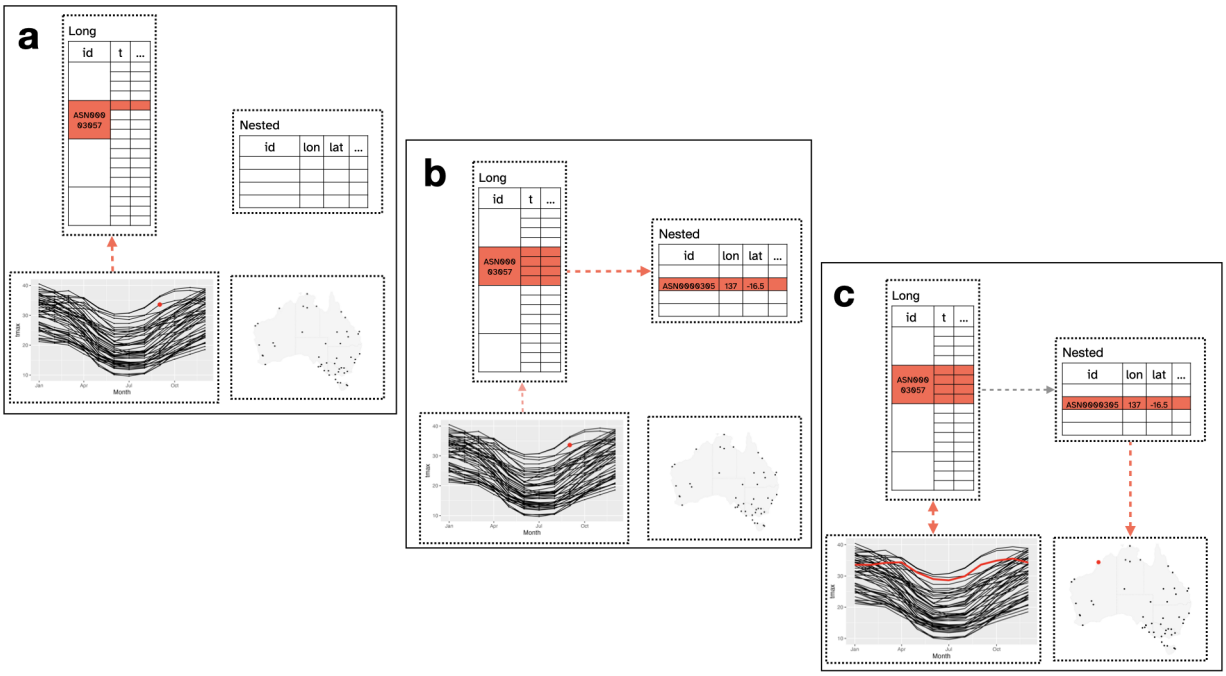

Figure 2: Linking between multiple plots. The line plots and the map are constructed from shared crosstalk objects (long and nested cubbles). When a point on the time series is selected, the corresponding row in the long cubble will be activated (a). This will link to all the rows with the same id in the long cubble and the row in the nested cubble with the same id (b). Both plots will be updated with the full line selected and the point highlighted on the map (c).