R Markdown (World Mapping for Abstracts)

Emma Chapman-Banks

2025-01-17

Part 1: Map Abstract Geolocation

Load necessary packages

```
library("tidyverse")
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library("sf")
```

```
## Linking to GEOS 3.11.0, GDAL 3.5.3, PROJ 9.1.0; sf_use_s2() is TRUE
```

```
library("rnaturalearth")
library("countrycode")
library("ggrepel")
```

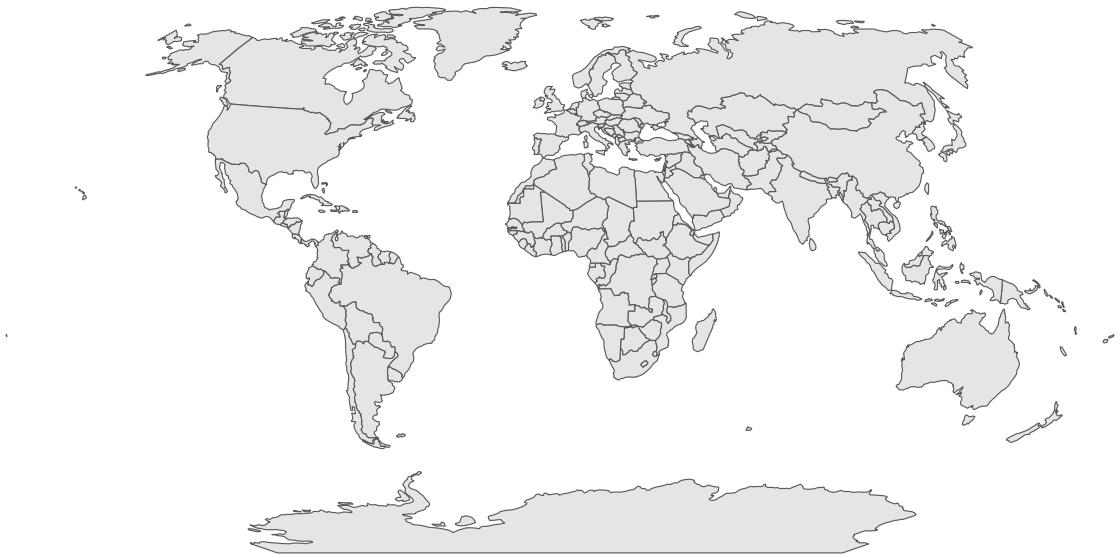
Get world data/change map projection (from Paula's code)

```
# Get world data
world <- ne_countries(scale = "small", returnclass = "sf")
world$iso_a3 <- ifelse(world$iso_a3 == "-99",
                       world$iso_a3_eh,
                       world$iso_a3)

world$iso_a3 <- ifelse(world$iso_a3 == "-99",
                       world$sov_a3,
                       world$iso_a3)

# Change map projection
```

```r
world %>%
  st_transform(crs = "+proj=wintri") %>%
  ggplot() +
  geom_sf() +
  coord_sf(datum = NA) + # no graticule
  theme_minimal()
```



Now, we will look at the data from the abstracts. Country data information is found in the llm_location column. When first originally running the code, the following warning came about:

"Warning: Some values were not matched unambiguously: England & Wales, England and Wales, England, UK, Europe, Kalutara District, Lausanne, New York metropolitan area, sub-Saharan Africa, Washington State"

To fix this, we need to manually map these locations. And these locations are added into the code chunk below (see lines 52-71). Subsaharan Africa and Europe are excluded because it is quite broad, but obviously, up to Paula and others to determine whether they want to include/exclude. Moreover, certain locations that are already valid ISO3 codes are excluded from being remapped and avoids redundant remapping.

```r
# Load new dataset with geolocations
abstracts_data <- read.csv("data/Combined_Location_With_Metadata.csv", stringsAsFactors = FALSE)

# Define manual mappings only for ambiguous single locations
manual_mapping <- list(
  "England and Wales" = "GBR",
  "England & Wales" = "GBR",
  "Fayoum, Egypt" = "EGY",
```

```
  "New York metropolitan area" = "USA",
  "Hebei Province, China" = "CHN",
  "Brasilia, Brazil" = "BRA",
  "Michigan, U.S.A.; New York City, U.S.A." = "USA",
  "Edmonton, Alberta, Canada" = "CAN",
  "South India" = "IND",
  "sub-Saharan Africa" = "unmapped",  # Exclude broad regions
  "southeastern United States" = "USA",
  "Kalutara District; Sri Lanka" = "LKA",
  "Lausanne" = "CHE",
  "Britain" = "GBR",
  "England, UK" = "GBR",
  "U.S." = "USA",
  "Mexico; Southern region of Mexico" = "MEX",
  "Washington State" = "USA",
  "Europe" = "unmapped",
  "Kalutara District" = "LKA"
)

# Process `llm_location` column
abstracts_data_clean <- abstracts_data %>%
  filter(!is.na(llm_location) & llm_location != "") %>%
  separate_rows(llm_location, sep = ";\\s*") %>%
  mutate(
    llm_location_resolved = recode(llm_location, !!!manual_mapping, .default = llm_location),
    Iso3 = ifelse(
      llm_location_resolved %in% c("BRA", "CAN", "CHE", "CHN", "EGY", "GBR", "IND", "LKA"), # Skip rema
      llm_location_resolved,
      countrycode::countrycode(
        sourcevar = llm_location_resolved,
        origin = "country.name",
        destination = "iso3c",
        warn = FALSE
      )
    )
  ) %>%
  mutate(Iso3 = ifelse(is.na(Iso3), "unmapped", Iso3))  # Mark unmatched values as "unmapped"

# Add a column to indicate when a location is mapped or unmapped
abstracts_data_clean <- abstracts_data_clean %>%
  mutate(Status = ifelse(Iso3 == "unmapped", "Unmapped", "Mapped"))

# Identify unmatched values for review
unmatched <- abstracts_data_clean %>%
  filter(Status == "Unmapped") %>%
  distinct(llm_location_resolved) %>%
  arrange(llm_location_resolved)

# Print unmatched values for verification. This is to check and confirm that there hasn't been any coun
print("Unmatched Values:")
```

```
## [1] "Unmatched Values:"
```

```r
print(unmatched)
```

```
## # A tibble: 1 x 1
##   llm_location_resolved
##   <chr>
## 1 unmapped
```

```r
# Group by ISO3 code and Status, include unmapped entries
abstracts_by_country <- abstracts_data_clean %>%
  group_by(Iso3, Status) %>%
  summarize(n_abstracts = n(), .groups = "drop") %>%
  mutate(n_abstracts_log = log(n_abstracts + 1))  # Add 1 to avoid log(0) issues
```

```r
# Merge with world map
world_with_abstracts <- world %>%
  left_join(abstracts_by_country, by = c("iso_a3" = "Iso3"))
```

```r
# Plot the world map
ggplot() +
  geom_sf(data = world, fill = "grey90", color = "grey") +
  geom_sf(data = world_with_abstracts, aes(fill = n_abstracts_log), color = NA) +
  scale_fill_gradient(low = "lightblue", high = "darkblue", na.value = "grey90",
                      name = "Log Abstracts") +
  theme_minimal() +
  theme(
    plot.title = element_text(face = "bold"),
    axis.text.x = element_blank(),
    legend.position = "bottom",
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 8)
  ) +
  labs(
    title = "Geographic Distribution of Abstracts",
    x = NULL, y = NULL
  )
```

Geographic Distribution of Abstracts