Loan word maps

Henning Schreiber1,✉, and Nikolas Gestrich2

March 17, 2025

Abstract

Text of abstract

1 Universität Hamburg  
2 Frobenius Institute

✉ Correspondence: [Henning Schreiber <janedoe@fosg.org>](mailto:janedoe@fosg.org)

Keywords: keyword 1; keyword 2; keyword 3

Highlights: These are the highlights.

# Creating language maps

Read location data from glottolog and Sahelia databases

# Read data  
langloc <- readr::read\_csv(here::here('analysis/data/raw\_data/glottologAfricanLanguages.csv')) |>  
   
 #retain only relevant variables  
 dplyr::select(glott\_code, latitude, longitude, LanName) |>   
   
# drop languages with unknown location  
 tidyr::drop\_na(longitude) |>   
  
 # turn latitude and longitude columns into geometry  
 sf::st\_as\_sf(coords = c("longitude","latitude"))  
  
# set crs  
sf::st\_crs(langloc) <- 4326

# read classified loan word data  
# Create a vector of file paths  
objs <- here::here("analysis", "data", "raw\_data", c("gold.csv",   
 "fonio.csv",  
 "millet.csv",  
 "salt.csv",  
 "iron.csv"))  
  
# read the csv files into a list of data frames  
data\_list <- purrr::map(objs, readr::read\_csv)  
  
# Add the location data to the word lists  
  
# Function to join location data to each data frame  
join\_geometry <- function(df) {  
 df |>   
 dplyr::left\_join(langloc, by = "glott\_code") |>   
 sf::st\_as\_sf() # Convert the result to an sf object if needed  
}  
  
# Apply the join function to each data frame in the list  
data\_list <- purrr::map(data\_list, join\_geometry)

The data contains certain duplicate entries. These need to be removed to avoid overly full plots

# Function to remove duplicates  
remove\_duplicates <- function(df\_list) {  
 df\_list <- lapply(df\_list, function(df) {  
 df <- df |>   
 dplyr::distinct(glott\_code, form, .keep\_all = TRUE)  
 return(df)  
 })  
 return(df\_list)  
}  
  
# Apply function  
data\_list <- remove\_duplicates(data\_list)

Function to make interactive leaflet maps

create\_leaflet\_map <- function(data\_sf, form\_column = "form", style\_column = "style", background = "colour") {  
   
 # Ensure the input data is an sf object and contains the required columns  
 if (!inherits(data\_sf, "sf")) {  
 stop("The input data must be an sf object.")  
 }  
 if (!style\_column %in% colnames(data\_sf)) {  
 stop(paste("The column", style\_column, "is not found in the data."))  
 }  
 if (!"glott\_code" %in% colnames(data\_sf)) {  
 stop("The column 'glott\_code' is not found in the data.")  
 }  
 if (!"LanName" %in% colnames(data\_sf)) {  
 stop("The column 'LanName' is not found in the data.")  
 }  
  
 # Filter out rows with NA values in the "style" column  
 data\_sf <- data\_sf |>  
 dplyr::filter(!is.na(!!rlang::sym(style\_column)))  
   
 # Set the tile provider based on the 'background' argument  
 tile\_provider <- switch(  
 background,  
 "colour" = leaflet::providers$Esri.WorldPhysical, # Default color tile  
 "bw" = leaflet::providers$Esri.WorldGrayCanvas, # Black-and-white tile  
 stop("Invalid value for 'background'. Choose 'colour' or 'bw'.")  
 )  
  
 # Define a color palette for the "style" column (using Dark2 palette from RColorBrewer)  
 color\_pal <- leaflet::colorFactor(palette = RColorBrewer::brewer.pal(8, "Dark2"), domain = data\_sf[[style\_column]])  
  
 # Add a new column with the corresponding colors based on the 'style' column  
 data\_sf <- data\_sf |>  
 dplyr::mutate(color = color\_pal(!!rlang::sym(style\_column))) # Dynamically reference the column  
  
 # Adjust overlapping points by slightly jittering their coordinates  
 # Initially apply fixed jitter for the map creation  
 jitter\_scale <- 0.5000 # Set the base jitter scale factor  
  
 data\_sf <- data\_sf |>  
 dplyr::mutate(  
 original\_lon = sf::st\_coordinates(geometry)[, 1], # Store original longitude  
 original\_lat = sf::st\_coordinates(geometry)[, 2], # Store original latitude  
 jittered\_lon = original\_lon + stats::rnorm(dplyr::n(), mean = 0, sd = jitter\_scale),  
 jittered\_lat = original\_lat + stats::rnorm(dplyr::n(), mean = 0, sd = jitter\_scale)  
 )  
  
 # Create the leaflet map  
 leaflet\_map <- leaflet::leaflet(data\_sf) |>  
 leaflet::addProviderTiles(tile\_provider) |>  
 leaflet::addLabelOnlyMarkers(  
 lng = ~jittered\_lon, # Use jittered longitude  
 lat = ~jittered\_lat, # Use jittered latitude  
 label = purrr::map(  
 glue::glue("<span style='color:{data\_sf$color}; font-weight: bold; font-size: 12px;'>{as.character(data\_sf[[form\_column]])}</span>"),  
 htmltools::HTML  
 ),  
 labelOptions = leaflet::labelOptions(  
 noHide = TRUE,  
 direction = 'top',  
 textOnly = TRUE  
 )  
 ) |>  
 leaflet::addMarkers(  
 lng = ~jittered\_lon,  
 lat = ~jittered\_lat,  
 popup = ~paste("<strong>Glottocode:</strong> ", glott\_code, "<br><strong>Language Name:</strong> ", LanName),  
 options = leaflet::markerOptions(opacity = 0)  
 )  
  
 # Add JavaScript to dynamically adjust jitter based on zoom level  
 leaflet\_map <- leaflet\_map |>  
 htmlwidgets::onRender(  
 "  
 function(el, map) {  
 // Function to apply jitter based on zoom level  
 function applyJitter() {  
 var zoomLevel = map.getZoom();  
 var jitterScale = 0.5 \* Math.pow(20, -zoomLevel); // Increase jitter at lower zoom levels  
  
 // Adjust the jitter of each marker  
 map.eachLayer(function(layer) {  
 if (layer instanceof L.Marker) {  
 var originalLat = layer.options.originalLat; // Access the original latitude  
 var originalLng = layer.options.originalLng; // Access the original longitude  
 var newLat = originalLat + (Math.random() - 0.5) \* jitterScale;  
 var newLng = originalLng + (Math.random() - 0.5) \* jitterScale;  
 layer.setLatLng([newLat, newLng]);  
 }  
 });  
 }  
  
 // Apply jitter on map zoom change  
 map.on('zoomend', function() {  
 applyJitter();  
 });  
  
 // Apply jitter initially  
 applyJitter();  
 }  
 ",  
 list(  
 originalLat = ~original\_lat,  
 originalLng = ~original\_lon  
 )  
 )  
   
 # Return the map object  
 return(leaflet\_map)  
}

Function to save a version of every map in the list of data

# Function to create and save leaflet maps for all data frames in data\_list  
save\_maps <- function(data\_list, form\_column = "form", background = "colour") {  
   
 # Use purrr::map2 to iterate over the list of data frames and generate maps  
 purrr::map2(data\_list, seq\_along(data\_list), ~ {  
 # Create the map using the create\_leaflet\_map function  
 map <- create\_leaflet\_map(.x, form\_column, background)  
   
 # Generate the filename based on the index or data frame name  
 filename <- paste0("map\_", .y, ".html")  
   
 # Save the map as an HTML file in the 'analysis/figures' folder  
 htmlwidgets::saveWidget(map, here::here("analysis/figures", filename))  
 })  
}

Save an html version of all objects

save\_maps(data\_list, background = "bw")

plot(rnorm(10))

|  |
| --- |
| Figure 1: A plot of random numbers |

[Figure 1](#fig-demo-plot) shows how we can have a caption and cross-reference for a plot. Note that figure label and cross-references must both be prefixed with fig-

Here is an example of inline code 3.14 in the middle of a sentence.

# Discussion

# Conclusion

# Acknowledgements

# References

### Colophon

This report was generated on 2025-03-17 16:42:37.470164 using the following computational environment and dependencies:

# which R packages and versions?  
if ("devtools" %in% installed.packages()) devtools::session\_info()

─ Session info ───────────────────────────────────────────────────────────────  
 setting value  
 version R version 4.4.2 (2024-10-31)  
 os macOS Sequoia 15.3.1  
 system x86\_64, darwin20  
 ui X11  
 language (EN)  
 collate en\_US.UTF-8  
 ctype en\_US.UTF-8  
 tz Europe/Berlin  
 date 2025-03-17  
 pandoc 3.2 @ /Applications/RStudio.app/Contents/Resources/app/quarto/bin/tools/x86\_64/ (via rmarkdown)  
 quarto 1.5.57 @ /Applications/RStudio.app/Contents/Resources/app/quarto/bin/quarto  
  
─ Packages ───────────────────────────────────────────────────────────────────  
 ! package \* version date (UTC) lib source  
 P bit 4.5.0.1 2024-12-03 [?] RSPM  
 P bit64 4.6.0-1 2025-01-16 [?] RSPM  
 P cachem 1.1.0 2024-05-16 [?] RSPM  
 class 7.3-22 2023-05-03 [2] CRAN (R 4.4.2)  
 P classInt 0.4-11 2025-01-08 [?] RSPM  
 P cli 3.6.4 2025-02-13 [?] RSPM  
 P crayon 1.5.3 2024-06-20 [?] RSPM  
 P DBI 1.2.3 2024-06-02 [?] RSPM  
 P devtools 2.4.5 2022-10-11 [?] RSPM  
 P digest 0.6.37 2024-08-19 [?] RSPM  
 P dplyr 1.1.4 2023-11-17 [?] RSPM  
 P e1071 1.7-16 2024-09-16 [?] RSPM  
 P ellipsis 0.3.2 2021-04-29 [?] RSPM  
 P evaluate 1.0.3 2025-01-10 [?] RSPM  
 P fastmap 1.2.0 2024-05-15 [?] RSPM  
 P fs 1.6.5 2024-10-30 [?] RSPM  
 P generics 0.1.3 2022-07-05 [?] RSPM  
 P glue 1.8.0 2024-09-30 [?] RSPM  
 P here 1.0.1 2020-12-13 [?] RSPM  
 P hms 1.1.3 2023-03-21 [?] RSPM  
 P htmltools 0.5.8.1 2024-04-04 [?] RSPM  
 P htmlwidgets 1.6.4 2023-12-06 [?] RSPM  
 P httpuv 1.6.15 2024-03-26 [?] RSPM  
 P jsonlite 1.9.1 2025-03-03 [?] RSPM  
 KernSmooth 2.23-24 2024-05-17 [2] CRAN (R 4.4.2)  
 P knitr 1.49 2024-11-08 [?] RSPM  
 P later 1.4.1 2024-11-27 [?] RSPM  
 P lifecycle 1.0.4 2023-11-07 [?] RSPM  
 P magrittr 2.0.3 2022-03-30 [?] RSPM  
 P memoise 2.0.1 2021-11-26 [?] RSPM  
 P mime 0.12 2021-09-28 [?] RSPM  
 P miniUI 0.1.1.1 2018-05-18 [?] RSPM  
 P pillar 1.10.1 2025-01-07 [?] RSPM  
 P pkgbuild 1.4.6 2025-01-16 [?] RSPM  
 P pkgconfig 2.0.3 2019-09-22 [?] RSPM  
 P pkgload 1.4.0 2024-06-28 [?] RSPM  
 P profvis 0.4.0 2024-09-20 [?] RSPM  
 P promises 1.3.2 2024-11-28 [?] RSPM  
 P proxy 0.4-27 2022-06-09 [?] RSPM  
 P purrr 1.0.4 2025-02-05 [?] RSPM  
 P R6 2.6.1 2025-02-15 [?] RSPM  
 P Rcpp 1.0.14 2025-01-12 [?] RSPM  
 P readr 2.1.5 2024-01-10 [?] RSPM  
 P remotes 2.5.0 2024-03-17 [?] RSPM  
 P rlang 1.1.5 2025-01-17 [?] RSPM  
 P rmarkdown 2.29 2024-11-04 [?] RSPM  
 P rprojroot 2.0.4 2023-11-05 [?] RSPM  
 P rstudioapi 0.17.1 2024-10-22 [?] RSPM  
 P s2 1.1.7 2024-07-17 [?] RSPM  
 P sessioninfo 1.2.3 2025-02-05 [?] RSPM  
 P sf 1.0-19 2024-11-05 [?] RSPM  
 P shiny 1.10.0 2024-12-14 [?] RSPM  
 P tibble 3.2.1 2023-03-20 [?] RSPM  
 P tidyr 1.3.1 2024-01-24 [?] RSPM  
 P tidyselect 1.2.1 2024-03-11 [?] RSPM  
 P tzdb 0.4.0 2023-05-12 [?] RSPM  
 P units 0.8-5 2023-11-28 [?] RSPM  
 P urlchecker 1.0.1 2021-11-30 [?] RSPM  
 P usethis 3.1.0 2024-11-26 [?] RSPM  
 P vctrs 0.6.5 2023-12-01 [?] RSPM  
 P vroom 1.6.5 2023-12-05 [?] RSPM  
 P withr 3.0.2 2024-10-28 [?] RSPM  
 P wk 0.9.4 2024-10-11 [?] RSPM  
 P xfun 0.51 2025-02-19 [?] RSPM  
 P xtable 1.8-4 2019-04-21 [?] RSPM  
 P yaml 2.3.10 2024-07-26 [?] RSPM  
  
 [1] /Users/nikolasgestrich/Library/Caches/org.R-project.R/R/renv/library/WAlingex-413b101e/macos/R-4.4/x86\_64-apple-darwin20  
 [2] /Library/Frameworks/R.framework/Versions/4.4-x86\_64/Resources/library  
  
 P ── Loaded and on-disk path mismatch.  
  
──────────────────────────────────────────────────────────────────────────────

The current Git commit details are:

# what commit is this file at?   
if ("git2r" %in% installed.packages() & git2r::in\_repository(path = ".")) git2r::repository(here::here())

Local: main /Users/nikolasgestrich/Documents/github/WAlingex  
Remote: main @ origin (https://github.com/nickgestrich/WAlingex.git)  
Head: [0f79853] 2025-03-17: update dependencies