Measuring Typological and Goegraphical Distances in R

Yingqi Jing

April 5, 2022

Contents

1 Alternative functions to ifelse & loop in a data.frame

1.1 Generalize `ifelse` to `case_when`

```
lang_df = data.frame(id = 1:4,
                     lang = c("Estonian", "Finnish", "Hungarian", "North Saami"))
```

1.2 Generalize `for` loop to `rowwise` operation

2 Typological similarities between Uralic languages

2.1 Function

```
typological_sim = function(data = ut_final, x, y){
  subdata = data %>%
    dplyr::select(-matches("subfamily|area", ignore.case = T)) %>%
    column_to_rownames(var = "Name")
  sim = sum(abs(as.vector(subdata[x, ]) == as.vector(subdata[y, ])))
  sim_p = sim/ncol(subdata)
  return(sim_p)
}
```

2.2 Data preprocessing

```
uratyp_df = read.csv("../Data/uratyp-1.1/cldf/values.csv")
lang_df = read.csv("../Data/uratyp-1.1/cldf/languages.csv")
ut_data = uratyp_df %>%
  inner_join(., lang_df, by = c("Language_ID" = "ID")) %>%
  dplyr::select(Name, Parameter_ID, Value, Subfamily) %>%
  filter(grepl("UT", Parameter_ID))
```

(1) Convert all data into binary (0, 1)

```
ut_wide = ut_data %>%
  mutate(Value = case_when(
    Value == "0" ~ 0L,
    Value == "1" ~ 1L,
    TRUE ~ NA_integer_ # convert all "?" into NA
  )) %>%
  pivot_wider(., names_from = Parameter_ID, values_from = Value)
```

(2) Remove all columns with missing values

```
ut_wide = ut_wide %>%
  select_if(function(x) !any(is.na(x)))
# alternatively, select_if(~ !any(is.na(.x)))
# select_if(~ sum(is.na(.x)) == 0)
# select(where(~ sum(is.na(.x)) == 0))
```

(3) Remove all constant columns

```
ut_final = ut_wide %>%
  remove_constant(.)
  # select_if(~ length(unique(.x)) > 1)
```

(4) Visualize data via heatmap

```
lang_sorted = ut_final %>%
  arrange(Subfamily) %>%
  pull(Name)
ut_final_long = ut_final %>%
  pivot_longer(., names_to = "feature", values_to = "value", -c("Name", "Subfamily")) %>%
  mutate(value = factor(value),
         Name = factor(Name, levels = lang_sorted))
ut_final_long %>%
  ggplot(., aes(feature, Name, fill = value)) +
  geom_tile() +
  scale_fill_manual(values = alpha(c("blue", "red"), 0.65)) +
  theme(axis.text.x = element_text(angle = 90, size = 6, hjust = 0),
        axis.ticks = element_blank()) +
  labs(x = NULL, y = NULL)
```

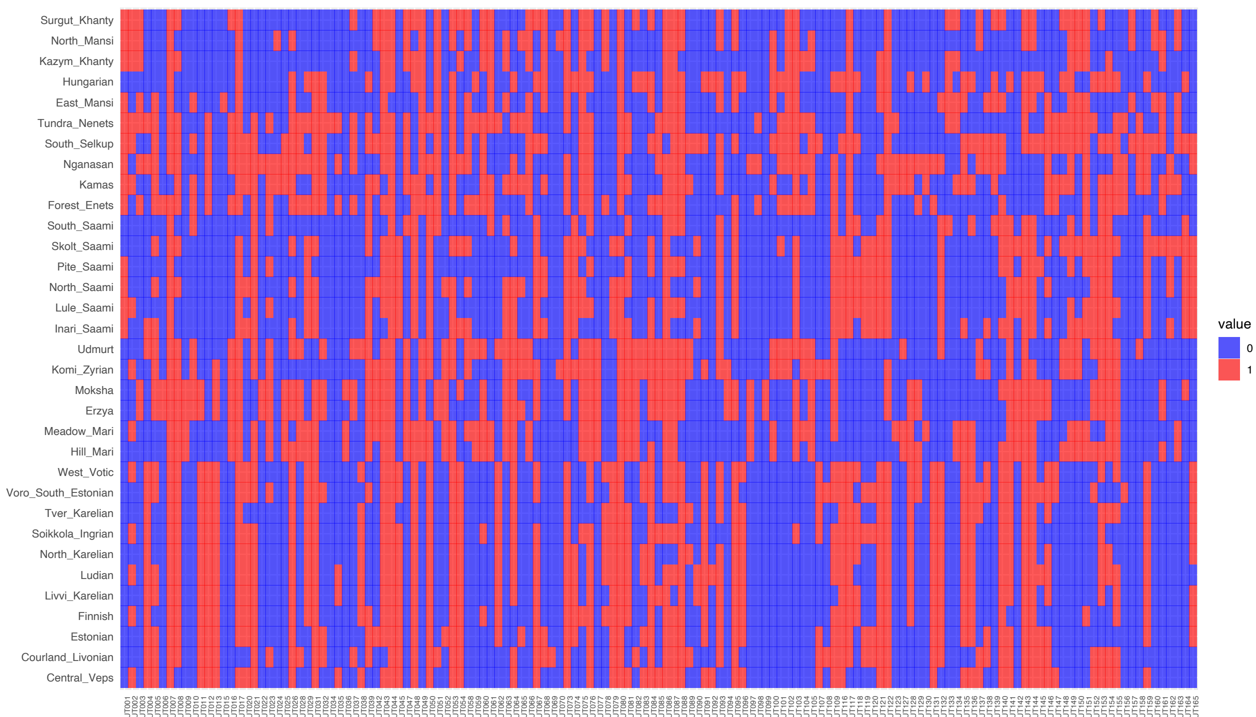
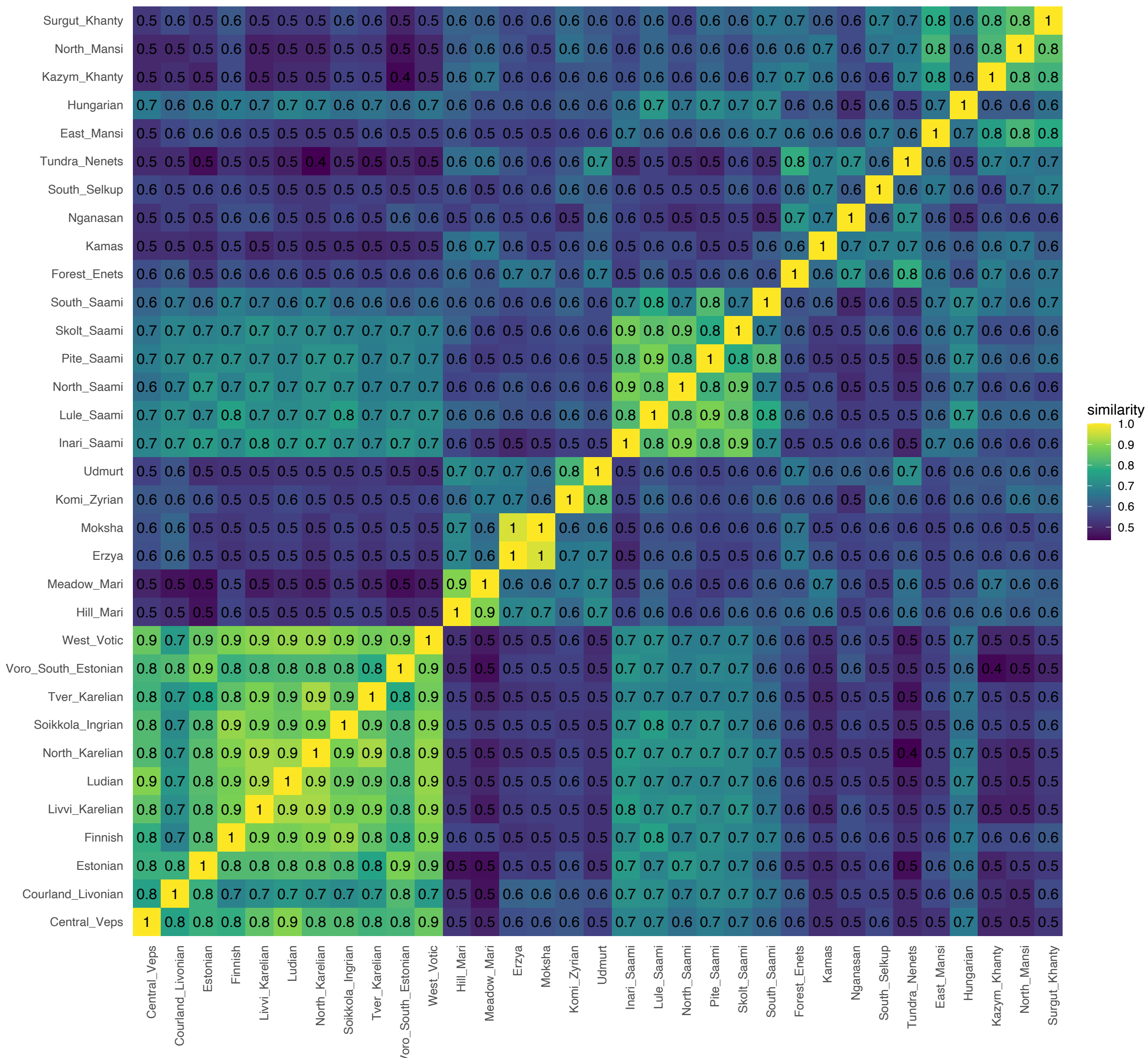


Figure 1: Overview of typological data in UT database

2.3 Calculating typological similarities

```
lgs = ut_final$Name
lgs_sim = expand.grid(lgs, lgs, stringsAsFactors = F) %>%
  rowwise() %>%
  mutate(similarity = typological_sim(ut_final, Var1, Var2))
```

```
lgs_sim_sorted = lgs_sim %>%
  mutate(Var1 = factor(Var1, levels = lang_sorted),
         Var2 = factor(Var2, levels = lang_sorted))
ggplot(lgs_sim_sorted, aes(Var1, Var2, fill = similarity)) +
  geom_tile() +
  geom_text(data = lgs_sim_sorted,
            mapping = aes(Var1, Var2,
                          label = round(similarity, digit = 1))) +
  scale_fill_continuous(type = "viridis") +
  labs(x = NULL, y = NULL) +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        axis.text.y = element_text(size = 9),
        axis.text.x = element_text(angle = 90, size = 9, hjust = 1),
        axis.ticks = element_blank(),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank()) +
  coord_fixed()
```


Surgut_Khanty 0.5 0.6 0.5 0.6 0.5 0.5 0.5 0.6 0.6 0.5 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.7 0.6 0.6 0.7 0.7 0.8 0.6 0.8 0.8 1
North_Mansi 0.5 0.5 0.5 0.6 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.6 0.6 0.6 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.6 0.7 0.7 0.8 0.6 0.8 1 0.8
Kazym_Khanty 0.5 0.5 0.5 0.6 0.5 0.5 0.5 0.5 0.5 0.4 0.5 0.6 0.7 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.7 0.6 0.6 0.6 0.7 0.8 0.6 1 0.8 0.8
Hungarian 0.7 0.6 0.6 0.7 0.7 0.7 0.7 0.6 0.7 0.6 0.7 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.7 0.7 0.7 0.7 0.6 0.6 0.5 0.6 0.5 0.7 1 0.6 0.6 0.6
East_Mansi 0.5 0.6 0.6 0.6 0.5 0.5 0.5 0.5 0.6 0.5 0.5 0.6 0.5 0.5 0.5 0.6 0.6 0.7 0.6 0.6 0.6 0.6 0.7 0.6 0.6 0.6 0.7 0.6 1 0.7 0.8 0.8 0.8
Tundra_Nenets 0.5 0.5 0.5 0.5 0.5 0.5 0.4 0.5 0.5 0.5 0.5 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.5 0.5 0.5 0.6 0.5 0.8 0.7 0.7 0.6 1 0.6 0.5 0.7 0.7 0.7
South_Selkup 0.6 0.5 0.6 0.6 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.6 0.5 0.6 0.6 0.6 0.6 0.6 0.5 0.5 0.5 0.6 0.6 0.6 0.7 0.6 1 0.6 0.7 0.6 0.6 0.7 0.7
Nganasan 0.5 0.5 0.5 0.6 0.6 0.5 0.5 0.6 0.5 0.6 0.6 0.5 0.6 0.5 0.6 0.5 0.6 0.6 0.5 0.5 0.5 0.5 0.5 0.7 0.7 1 0.6 0.7 0.6 0.5 0.6 0.6 0.6
Kamas 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.6 0.7 0.6 0.5 0.6 0.6 0.5 0.6 0.6 0.5 0.5 0.6 0.6 1 0.7 0.7 0.7 0.6 0.6 0.6 0.7 0.6
Forest_Enets 0.6 0.6 0.5 0.6 0.6 0.6 0.5 0.6 0.5 0.6 0.6 0.6 0.6 0.7 0.7 0.6 0.7 0.5 0.6 0.5 0.6 0.6 0.6 1 0.6 0.7 0.6 0.8 0.6 0.6 0.7 0.6 0.7
South_Saami 0.6 0.7 0.6 0.7 0.7 0.6 0.7 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.8 0.7 0.8 0.7 1 0.6 0.6 0.5 0.6 0.5 0.7 0.7 0.7 0.6 0.7
Skolt_Saami 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.6 0.6 0.5 0.6 0.6 0.6 0.9 0.8 0.9 0.8 1 0.7 0.6 0.5 0.5 0.6 0.6 0.6 0.7 0.6 0.6 0.6
Pite_Saami 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.6 0.5 0.5 0.6 0.6 0.5 0.8 0.9 0.8 1 0.8 0.8 0.6 0.5 0.5 0.5 0.5 0.6 0.7 0.6 0.6 0.6
North_Saami 0.6 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.6 0.6 0.6 0.6 0.6 0.6 0.9 0.8 1 0.8 0.9 0.7 0.5 0.6 0.5 0.5 0.5 0.6 0.7 0.6 0.6 0.6
Lule_Saami 0.7 0.7 0.7 0.8 0.7 0.7 0.7 0.8 0.7 0.7 0.7 0.6 0.6 0.6 0.6 0.6 0.6 0.8 1 0.8 0.9 0.8 0.8 0.6 0.6 0.5 0.5 0.5 0.6 0.7 0.6 0.6 0.6
Inari_Saami 0.7 0.7 0.7 0.7 0.8 0.7 0.7 0.7 0.7 0.7 0.7 0.6 0.5 0.5 0.5 0.5 0.5 1 0.8 0.9 0.8 0.9 0.7 0.5 0.5 0.6 0.6 0.5 0.7 0.6 0.6 0.6 0.6
Udmurt 0.5 0.6 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.7 0.7 0.7 0.6 0.8 1 0.5 0.6 0.6 0.5 0.6 0.6 0.7 0.6 0.6 0.6 0.7 0.6 0.6 0.6 0.6 0.6
Komi_Zyrian 0.6 0.6 0.6 0.5 0.5 0.6 0.5 0.5 0.5 0.5 0.6 0.6 0.7 0.7 0.6 1 0.8 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6
Moksha 0.6 0.6 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.7 0.6 1 1 0.6 0.6 0.5 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.6 0.6 0.6 0.5 0.6 0.6 0.5 0.6
Erzya 0.6 0.6 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.7 0.6 1 1 0.7 0.7 0.5 0.6 0.6 0.5 0.5 0.6 0.7 0.6 0.5 0.6 0.6 0.5 0.6 0.6 0.6 0.6
Meadow_Mari 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.9 1 0.6 0.6 0.7 0.7 0.5 0.6 0.6 0.5 0.6 0.6 0.6 0.7 0.6 0.5 0.6 0.5 0.6 0.7 0.6 0.6
Hill_Mari 0.5 0.5 0.5 0.6 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1 0.9 0.7 0.7 0.6 0.7 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6
West_Votic 0.9 0.7 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 1 0.5 0.5 0.5 0.5 0.6 0.5 0.7 0.7 0.7 0.7 0.7 0.6 0.6 0.5 0.6 0.5 0.5 0.5 0.7 0.5 0.5 0.5
Voro_South_Estonian 0.8 0.8 0.9 0.8 0.8 0.8 0.8 0.8 0.8 1 0.9 0.5 0.5 0.5 0.5 0.5 0.5 0.7 0.7 0.7 0.7 0.7 0.6 0.6 0.5 0.6 0.5 0.5 0.5 0.6 0.4 0.5 0.5
Tver_Karelian 0.8 0.7 0.8 0.8 0.9 0.9 0.9 0.9 1 0.8 0.9 0.5 0.5 0.5 0.5 0.5 0.5 0.7 0.7 0.7 0.7 0.7 0.6 0.5 0.5 0.5 0.5 0.5 0.6 0.7 0.5 0.5 0.6
Soikkola_Ingrian 0.8 0.7 0.8 0.9 0.9 0.9 0.9 1 0.9 0.8 0.9 0.5 0.5 0.5 0.5 0.5 0.5 0.7 0.8 0.7 0.7 0.7 0.6 0.6 0.5 0.6 0.5 0.5 0.5 0.6 0.5 0.5 0.6
North_Karelian 0.8 0.7 0.8 0.9 0.9 0.9 1 0.9 0.9 0.8 0.9 0.5 0.5 0.5 0.5 0.5 0.5 0.7 0.7 0.7 0.7 0.7 0.7 0.5 0.5 0.5 0.5 0.4 0.5 0.7 0.5 0.5 0.5
Ludian 0.9 0.7 0.8 0.9 0.9 1 0.9 0.9 0.9 0.8 0.9 0.5 0.5 0.5 0.5 0.6 0.5 0.7 0.7 0.7 0.7 0.7 0.6 0.6 0.5 0.5 0.5 0.5 0.5 0.7 0.5 0.5 0.5
Livvi_Karelian 0.8 0.7 0.8 0.9 1 0.9 0.9 0.9 0.9 0.8 0.9 0.5 0.5 0.5 0.5 0.5 0.5 0.8 0.7 0.7 0.7 0.7 0.7 0.6 0.5 0.6 0.5 0.5 0.5 0.7 0.5 0.5 0.5
Finnish 0.8 0.7 0.8 1 0.9 0.9 0.9 0.9 0.8 0.8 0.9 0.6 0.5 0.5 0.5 0.5 0.5 0.7 0.8 0.7 0.7 0.7 0.7 0.6 0.5 0.6 0.6 0.5 0.6 0.7 0.6 0.6 0.6
Estonian 0.8 0.8 1 0.8 0.8 0.8 0.8 0.8 0.8 0.9 0.9 0.5 0.5 0.5 0.5 0.6 0.5 0.7 0.7 0.7 0.7 0.7 0.6 0.5 0.5 0.5 0.6 0.5 0.6 0.6 0.5 0.5 0.5
Courland_Livonian 0.8 1 0.8 0.7 0.7 0.7 0.7 0.7 0.7 0.8 0.7 0.5 0.5 0.6 0.6 0.6 0.6 0.7 0.7 0.7 0.7 0.7 0.7 0.6 0.5 0.5 0.5 0.5 0.6 0.6 0.5 0.5 0.6
Central_Veps 1 0.8 0.8 0.8 0.8 0.9 0.8 0.8 0.8 0.8 0.9 0.5 0.5 0.6 0.6 0.6 0.5 0.7 0.7 0.6 0.7 0.7 0.6 0.6 0.5 0.5 0.6 0.5 0.5 0.7 0.5 0.5 0.5
Central_Veps Courland_Livonian Estonian Finnish Livvi_Karelian Ludian North_Karelian Soikkola_Ingrian Tver_Karelian Voro_South_Estonian West_Votic Hill_Mari Meadow_Mari Erzya Moksha Komi_Zyrian Udmurt Inari_Saami Lule_Saami North_Saami Pite_Saami Skolt_Saami South_Saami Forest_Enets Kamas Nganasan South_Selkup Tundra_Nenets East_Mansi Hungarian Kazym_Khanty North_Mansi Surgut_Khanty
similarity
1.0
0.9
0.8
0.7
0.6
0.5

2.4 Exercises

(1) Pls add the information of typological areas (*uratyp-1.1/cldf/parameters.csv*) to UT dataset. Note: you can use the preprocessed UT data **ut_final_long**.

```
feat_areas = read.csv("../Data/uratyp-1.1/cldf/parameters.csv")
feat_areas = feat_areas %>%
  dplyr::select(ID, Area)
ut_areas = ut_final_long %>%
  inner_join(., feat_areas, by = c("feature" = "ID"))
```

(2) Pls calculate the typological similarities between languages across typological areas (phonology, morphology and syntax) in UT dataset, and plot them as heatmaps separately. Note: you can remove the lexicon features, and use `facet_wrap` function to create subpanels.

```
ut_area_sim = ut_areas %>%
  filter(Area %in% c("Phonology", "Morphology", "Syntax")) %>%
  split(.$Area) %>%
  map_dfr(., ~{subdata = .x %>% spread(., key = feature, value = value)
              expand.grid(lgs, lgs, stringsAsFactors = F) %>%
                rowwise() %>%
                mutate(similarity = typological_sim(subdata, Var1, Var2)) %>%
                mutate(Area = subdata$Area[1])})
```

```
ut_area_sim_sorted = ut_area_sim %>%
  mutate(Var1 = factor(Var1, levels = lang_sorted),
         Var2 = factor(Var2, levels = lang_sorted),
         Area = factor(Area, levels = c("Phonology", "Morphology", "Syntax")))
ut_area_sim_sorted %>%
  ggplot(., aes(Var1, Var2, fill = similarity)) +
  geom_tile() +
  # geom_text(data = lgs_sim_sorted,
  #           mapping = aes(Var1, Var2,
  #                         label = round(similarity, digit = 1))) +
  scale_fill_continuous(type = "viridis") +
  facet_wrap(~Area) +
  # scale_x_discrete(position = "bottom") +
  labs(x = NULL, y = NULL) +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        axis.text.y = element_text(size = 9),
        axis.text.x = element_text(angle = 90, size = 9, hjust = 1),
        axis.ticks = element_blank(),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank()) +
  coord_fixed()
```

3 Geographical distances between languages

3.1 Function

```
geographical_dist = function(data = lang_geo, x = lang1, y = lang2){
  lang1_location = data[x, ]
  lang2_location = data[y, ]
  return(distHaversine(lang1_location, lang2_location)/1000)
}
```

3.2 Measuring geographical distances

```
lang_geo = lang_df %>%
  dplyr::select(Name, Longitude, Latitude) %>%
  column_to_rownames(var = "Name")
```

```
geo_dist = expand.grid(lgs, lgs, stringsAsFactors = F) %>%
  rowwise() %>%
  mutate(distance = geographical_dist(data = lang_geo,
                                      x = Var1,
                                      y = Var2)) %>%
  ungroup %>%
  mutate(dist_scaled = distance/max(distance))
geo_dist_sorted = geo_dist %>%
  mutate(Var1 = factor(Var1, levels = lang_sorted),
         Var2 = factor(Var2, levels = lang_sorted))
```

```
geo_dist_sorted %>%
  ggplot(., aes(Var1, Var2, fill = distance)) +
  geom_tile() +
  geom_text(data = geo_dist_sorted,
            mapping = aes(Var1, Var2,
                          label = round(dist_scaled, digit = 1))) +
  scale_fill_continuous(type = "viridis", direction = -1) +
  labs(x = NULL, y = NULL) +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        axis.text.y = element_text(size = 9),
        axis.text.x = element_text(angle = 90, size = 9, hjust = 1),
        axis.ticks = element_blank(),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank()) +
  coord_fixed()
```

```
# lgs_sim = combn(lgs, 2) %>%
#   t() %>%
#   as.data.frame() %>%
#   rowwise() %>%
#   mutate(similarity = mutual_sim(ut_final, V1, V2))
```

```
# library(reshape2)
# get_upper_tri <- function(mat){
#     mat[lower.tri(mat)]<- NA
#     return(mat)
```

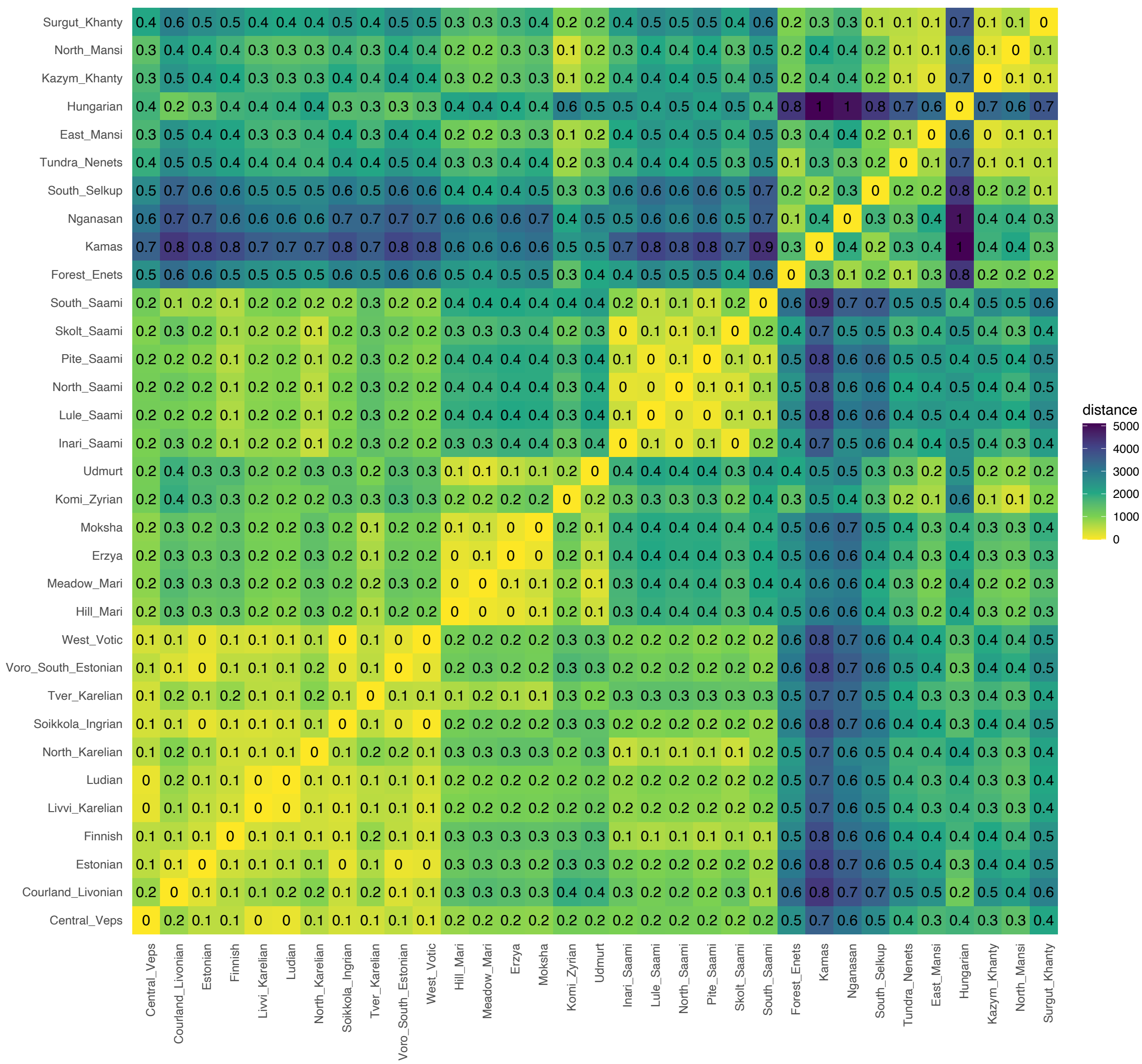


Figure 2: Geographical distances (scaled) between Uralic languages

```
# }
# sim_mat = ncol(ut_df) - as.matrix(dd)
# sim_mat = get_upper_tri(sim_mat)
# melted_mat <- melt(sim_mat, na.rm = TRUE)
# ggplot(data = melted_cormat, aes(Var2, Var1, fill = value)) +
#  geom_tile(color = "white") +
#   scale_fill_continuous(type = "viridis") +
#   theme_minimal() +
#  theme(axis.text.x = element_text(angle = 90, size = 9, hjust = 1))
```

```
# A <- c(1,4,5,6,1)
# B <- c(4,2,5,6,7)
# C <- c(3,4,2,4,6)
# D <- c(2,5,1,4,6)
# E <- c(6,7,8,9,1)
#
# df <- data.frame(A,B,C,D,E)
#
# CorMat <- cor(df[ ,c("A","B","C","D","E")])
#
# get_upper_tri <- function(CorMat){
#   CorMat[upper.tri(CorMat)]<- NA
#   return(CorMat)
# }
#
# get_lower_tri <- function(CorMat){
#   CorMat[lower.tri(CorMat)]<- NA
#   return(CorMat)
# }
#
# reorder <- function(CorMat){
#   dd <- as.dist((1-CorMat)/2)
#   hc <- hclust(dd)
#   CorMar <- CorMat[hc$order, hc$order]
# }
#
# library(reshape2)
#
# CorMat <- reorder(CorMat)
# upper_tri <- get_upper_tri(CorMat)
# lower_tri <- get_lower_tri(CorMat)
# meltNum <- melt(lower_tri, na.rm = T)
# meltColor <- melt(upper_tri, na.rm = T)
# library(tidyverse)
#  ggplot() +
#   labs(x = NULL, y = NULL) +
#   geom_tile(data = meltColor,
#             mapping = aes(Var2, Var1,
#                           fill = value)) +
#   geom_text(data = meltNum,
#             mapping = aes(Var2, Var1,
#                           label = round(value, digit = 2))) +
#   scale_x_discrete(position = "top") +
#   # scale_fill_gradient(low = "white", high = "firebrick4",
```

```
#   #                      limit = c(-1,1), name = "Pearson\nCorrelation") +
#   theme(plot.title = element_text(hjust = 0.5, face = "bold"),
#         panel.grid.major = element_blank(),
#         panel.grid.minor = element_blank(),
#         panel.background = element_blank()) +
#   coord_fixed()
```

```
# dd = dist(ut_df, "manhattan") # calculate the absolute differences between langs
# plot(hclust(dd))

# pheat = pheatmap(as.matrix(ut_final),
#          legend = F,
#          cluster_cols = F)
```