Computing network measures and plotting the networks

TAD

2026-05-11

Code Sample: R

This file produces time series network metrices for unique and reproducible cultural product and non-cultural product trade networks for over the year 1995 to 2023. Then it plots network snapshots for all three product categories for the initial, middle and the ending year of the study period.

```
packages <- c("igraph", "wdnet", "dplyr", "plm","patchwork", "ggraph", "tidygraph", "here")
invisible(lapply(packages, function(pkg) {suppressPackageStartupMessages(library(pkg, charact
er.only = TRUE))}))
```

```
u <- read.csv(here("data", "cleaned", "unique_edgelist.csv"), header = TRUE,
              as.is = TRUE, stringsAsFactors = FALSE)
r <- read.csv(here("data", "cleaned", "reproducible_edgelist.csv"), header = TRUE,
              as.is = TRUE, stringsAsFactors = FALSE)
t <- read.csv(here("data", "cleaned", "total_edgelist.csv"), header = TRUE,
              as.is = TRUE, stringsAsFactors = FALSE)
```

```
degree_computation <- function(g1, alpha = 0.5) {outdeg <- centrality(g1, measure = "degree",
                      degree.control = list(alpha = alpha, mode = "out"))
  indeg <- centrality(g1,measure = "degree",degree.control = list(alpha = alpha,mode = "in"))
  CC <- clustcoef(g1, method = "Clemente")$total$localcc
  final_df <- data.frame(node = g1$node.attr$name,alpha = alpha,outdegree = outdeg,indegree =
indeg, clustering = CC)
  return(final_df)}
```

```
centrality_function <- function(m, year) {
  ycol <- paste0("X", year)
  m_y <- m[, c("iso_o", "iso_d", ycol)]
  m_y <- subset(m_y, m_y[[ycol]] > 0)
  net <- igraph::graph_from_data_frame(m_y,directed = TRUE)
  igraph::E(net)$weight <- igraph::edge_attr(net, ycol)
  g <- igraph_to_wdnet(net)
  deg <- degree_computation(g)
  deg_df <- deg %>%
    dplyr::select(-outdegree.name, -indegree.name) %>%dplyr::rename(country = node,out_deg =
outdegree.degree,in_deg= indegree.degree)
  n_nodes <- igraph::vcount(net)
  n_edges <- igraph::ecount(net)
  dens <- igraph::edge_density(net, loops = FALSE)
  recip <- igraph::reciprocity(net)
  trans <- igraph::transitivity(net,type = "global")
  avg_path <- suppressWarnings(igraph::mean_distance(net,directed = TRUE))
  deg_df <- deg_df %>% dplyr::mutate(year = year,n_nodes = n_nodes,n_edges = n_edges,density
= dens,
  reciprocity = recip,transitivity = trans,avg_path_length = avg_path)
   return(list(year = year,network = net,wdnet = g,measures = deg_df
  ))
}
```

```
nets_u <- lapply(1995:2023,function(y) centrality_function(u, y))
names(nets_u) <- 1995:2023

nets_r <- lapply(1995:2023,function(y) centrality_function(r, y))
names(nets_r) <- 1995:2023

nets_t <- lapply(1995:2023,function(y) centrality_function(t, y))
names(nets_t) <- 1995:2023
```

```
panel_u <- dplyr::bind_rows(lapply(nets_u, function(x) x$measures),.id = "year")
panel_r <- dplyr::bind_rows(lapply(nets_r, function(x) x$measures),.id = "year")
panel_t <- dplyr::bind_rows(lapply(nets_t, function(x) x$measures),.id = "year")
```

```
write.csv(panel_u, here("data", "cleaned", "network_metrics_u.csv"), row.names = FALSE)
write.csv(panel_r, here("data", "cleaned", "network_metrics_r.csv"), row.names = FALSE)
write.csv(panel_t, here("data", "cleaned", "network_metrics_t.csv"), row.names = FALSE)
```

Plotting the networks

```
# Selecting yearly snapshots
u1995 <- nets_u[["1995"]]$network
u2009 <- nets_u[["2009"]]$network
u2023 <- nets_u[["2023"]]$network


r1995 <- nets_r[["1995"]]$network
r2009 <- nets_r[["2009"]]$network
r2023 <- nets_r[["2023"]]$network

t1995 <- nets_t[["1995"]]$network
t2009 <- nets_t[["2009"]]$network
t2023 <- nets_t[["2023"]]$network
```

```
# Making a list of networks
nets <- list(
  u1995 = nets_u[["1995"]]$network,
  u2009 = nets_u[["2009"]]$network,
  u2023 = nets_u[["2023"]]$network,

  r1995 = nets_r[["1995"]]$network,
  r2009 = nets_r[["2009"]]$network,
  r2023 = nets_r[["2023"]]$network,

  t1995 = nets_t[["1995"]]$network,
  t2009 = nets_t[["2009"]]$network,
  t2023 = nets_t[["2023"]]$network
)
```

```
# Filtering the networks based on edge weight. I am keeping only the top 5% of the edges.
filter_network <- function(g, p = 0.95) {
  thr <- quantile(E(g)$weight, probs = p, na.rm = TRUE)
  g <- delete_edges(g, E(g)[weight < thr])
  g <- delete_vertices(g, V(g)[degree(g) == 0])
  V(g)$strength <- strength(g, mode = "all", weights = E(g)$weight)
  return(g)
}
```

```
nets_filtered <- lapply(nets, filter_network, p = 0.95)

# Defining a function to remove the edges those were with the nodes filtered out
define_hubs <- function(g) {
  cutoff <- quantile(V(g)$strength, 0.90, na.rm = TRUE)
  V(g)$label <- ifelse(V(g)$strength >= cutoff, V(g)$name, NA)
  return(g)
}
```

```
nets_filtered <- lapply(nets_filtered, define_hubs)
```

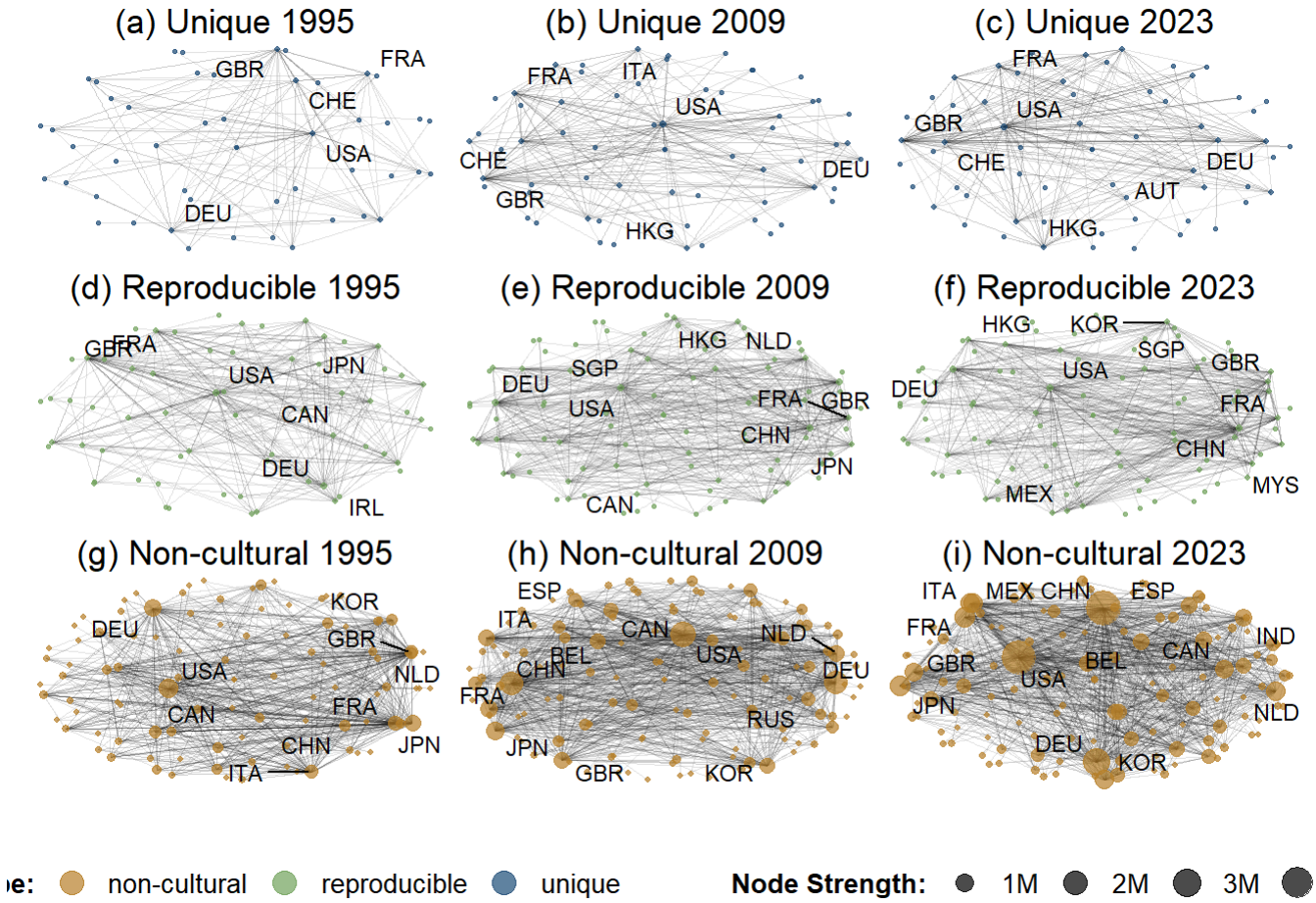
```
# Selecting filtered networks
u1995_f <- nets_filtered[["u1995"]]
u2009_f <- nets_filtered[["u2009"]]
u2023_f <- nets_filtered[["u2023"]]

r1995_f <- nets_filtered[["r1995"]]
r2009_f <- nets_filtered[["r2009"]]
r2023_f <- nets_filtered[["r2023"]]

t1995_f <- nets_filtered[["t1995"]]
t2009_f <- nets_filtered[["t2009"]]
t2023_f <- nets_filtered[["t2023"]]
```

For visualization purposes, I am using total weighted degree centrality (in+out strength) to capture overall network prominence, while directional centrality measures are reserved for econometric analysis

```
all_strengths <- unlist(lapply(nets_filtered, function(g) strength(g, mode = "all", weights =
E(g)$weight)))
all_weights   <- unlist(lapply(nets_filtered, function(g) E(g)$weight))

# 1. Clear, consistent named vector
type_colors <- c(unique = "#1F4E79", reproducible = "#74a55f", `non-cultural` = "#B8812A")

plot_network <- function(g, title_text, type_label) {
  V(g)$strength <- strength(g, mode = "all", weights = E(g)$weight)
  cutoff        <- quantile(V(g)$strength, 0.90, na.rm = TRUE)
  V(g)$label    <- ifelse(V(g)$strength >= cutoff, V(g)$name, NA)

  ggraph(g, layout = "on_sphere") +
    geom_edge_link(aes(width = weight), alpha = 0.1, show.legend = FALSE) +
    # Pulls color directly using the exact string key matching type_colors
    geom_node_point(aes(size = strength), color = type_colors[type_label], alpha = 0.7, show.
legend = FALSE) +
    geom_node_text(aes(label = label), repel = TRUE, size = 3, na.rm = TRUE) +

    scale_size(range = c(0.6, 6), limits = range(all_strengths)) + # Kept at c(0.6, 6)
    scale_edge_width(range = c(0.1, 1.5), limits = range(all_weights)) +
    ggtitle(title_text) +
    theme_void() +
    theme(
      plot.title = element_text(hjust = 0.5),
      legend.position = "none"
    )
}
```

```
# Generating subplots
p_u1995 <- plot_network(u1995_f, "(a) Unique 1995", "unique")
p_u2009 <- plot_network(u2009_f, "(b) Unique 2009", "unique")
p_u2023 <- plot_network(u2023_f, "(c) Unique 2023", "unique")

p_r1995 <- plot_network(r1995_f, "(d) Reproducible 1995", "reproducible")
p_r2009 <- plot_network(r2009_f, "(e) Reproducible 2009", "reproducible")
p_r2023 <- plot_network(r2023_f, "(f) Reproducible 2023", "reproducible")

# FIX: Passed "non-cultural" with the hyphen so it extracts the correct hex code
p_t1995 <- plot_network(t1995_f, "(g) Non-cultural 1995", "non-cultural")
p_t2009 <- plot_network(t2009_f, "(h) Non-cultural 2009", "non-cultural")
p_t2023 <- plot_network(t2023_f, "(i) Non-cultural 2023", "non-cultural")
```

```
# Generating footer for the legend
p_footer <- ggplot(data.frame(x = 1:3, y = 1:3, type = names(type_colors), strength = c(1e6,
3e6, 5e6)),
                   aes(x = x, y = y, color = type, size = strength)) +
  geom_point(alpha = 0) +
  scale_color_manual(values = type_colors, name = "Network Type:") +
  scale_size(range = c(0.6, 6), limits = range(all_strengths), name = "Node Strength:",
             labels = scales::label_number(scale_cut = scales::cut_short_scale())) +
  theme_void() +
  guides(color = guide_legend(nrow = 1, order = 1, override.aes = list(alpha = 0.7, size =
4)),
    size  = guide_legend(nrow = 1, order = 2, override.aes = list(alpha = 0.7))) +
  theme(legend.position     = "bottom",
        legend.box          = "horizontal",
        legend.direction    = "horizontal",
        legend.title        = element_text(size = 10, face = "bold"),
        legend.text         = element_text(size = 10),
        legend.spacing.x    = unit(1.5, "cm"),
        legend.box.spacing  = unit(0.1, "cm"))

network_grid <- wrap_plots(p_u1995, p_u2009, p_u2023,
                           p_r1995, p_r2009, p_r2023,
                           p_t1995, p_t2009, p_t2023,
                           ncol = 3, byrow = TRUE)
figure2 <- network_grid / p_footer +
  plot_layout(heights = c(12, 1))

print(figure2)
```




```
ggsave("../results/Fig2.tiff",figure2,width = 12,height = 12,dpi = 600)
```