```r
# --------------------------
# Sample Script: Undirected DTW Analysis for Individuals and Groups
# Load required R packages
# --------------------------

library(tidyverse)
library(parallelDist)
library(reshape2)
library(qgraph)
library(broom)

# --------------------------
# Create 3 example patient datasets (5 items, EDE_A to EDE_E)
# --------------------------

Patient1 <- tibble(time = 1:10,
  EDE_A = c(2, 3, 4, 5, 5, 4, 3, 4, 5, 6),
  EDE_B = c(1, 2, 5, 4, 5, 5, 4, 5, 5, 6),
  EDE_C = c(0, 1, 2, 2, 3, 2, 3, 4, 5, 5),
  EDE_D = c(2, 2, 3, 3, 4, 3, 4, 3, 4, 5),
  EDE_E = c(0, 1, 1, 1, 2, 1, 0, 1, 2, 3))

Patient2 <- tibble(time = 1:10,
  EDE_A = c(1, 2, 3, 2, 4, 5, 5, 6, 5, 4),
  EDE_B = c(2, 1, 3, 3, 5, 5, 4, 5, 4, 3),
  EDE_C = c(0, 0, 1, 2, 2, 2, 2, 3, 2, 1),
  EDE_D = c(0, 1, 1, 2, 2, 3, 2, 3, 2, 1),
  EDE_E = c(5, 0, 6, 2, 5, 1, 6, 0, 5, 2))

Patient3 <- tibble(time = 1:10,
  EDE_A = c(3, 2, 4, 5, 4, 5, 6, 5, 4, 3),
  EDE_B = c(2, 3, 3, 5, 4, 5, 5, 6, 5, 4),
  EDE_C = c(1, 2, 3, 3, 4, 4, 3, 4, 3, 2),
  EDE_D = c(1, 2, 2, 3, 4, 3, 3, 4, 4, 2),
  EDE_E = c(6, 1, 5, 2, 0, 6, 2, 5, 1, 6))

# --------------------------
# Function to calculate undirected DTW distances
# --------------------------

# Function to calculate undirected DTW distance matrix
calc_undirected_matrix <- function(db, timewindow = 1) {
  no_interpolations <- 5
  # Interpolate time series to minimize boundary mismatches
  interpolated_data <- map(db, ~ approx(.x, n = nrow(db) * no_interpolations)$y) %>%
    bind_cols()
  # Calculate pairwise undirected DTW distances
  distance_matrix <- parDist(t(interpolated_data), method = "dtw",
                             window.type = "sakoechiba",
                             step.pattern = "symmetric2",
                             norm.method = "n",
                             window.size = timewindow * no_interpolations) %>%
    as.matrix()
  colnames(distance_matrix) <- colnames(db)
  rownames(distance_matrix) <- colnames(db)
  return(distance_matrix)
}

# --------------------------
# Visualize undirected DTW for one patient
# --------------------------

# Remove 'time' column and calculate undirected DTW
distance1_undirected <- calc_undirected_matrix(Patient3[-1])

# Visualize the inverse of the distance matrix (stronger links appear stronger)
qgraph(1 / (distance1_undirected + 0.1),
       layout = "spring",
       threshold = 1) # Arrow only if directed distance is greater than 1)
# --------------------------
# Visualize the undirected DTW network for the group of 4 patients
# --------------------------

# The base R function scale(x) returns a matrix, not a simple numeric vector
```

```r
scale2 <- function(x, na.rm = TRUE) (x - mean(x, na.rm = na.rm)) / sd(x, na.rm)

# Combine 9 simulated individual patient datasets and transform into long format
tidy_patients <-  bind_rows(
  Patient1 %>% mutate(id = 1),
  Patient2 %>% mutate(id = 2),
  Patient3 %>% mutate(id = 3),
  Patient1 %>% mutate(id = 4),
  Patient2 %>% mutate(id = 5),
  Patient3 %>% mutate(id = 6),
  Patient1 %>% mutate(id = 7),
  Patient2 %>% mutate(id = 8),
  Patient3 %>% mutate(id = 9)) %>%
  # Reshape from wide (EDE_A, EDE_B, EDE_C) to long format (item, score)
  pivot_longer(cols = starts_with("EDE_"), names_to = "item", values_to = "score") %>%
  # Group-level standardize scores within each item
  group_by(item) %>%
  mutate(score = scale2(score)) %>%
  ungroup()

undirected1 <- tidy_patients %>%
  pivot_wider(names_from = item, values_from = score) %>%
  select(-time) %>%
  group_by(id) %>%
  nest() %>%
  mutate(distance = map(data, ~ calc_undirected_matrix(.x, timewindow = 1))) %>%
  ungroup() %>%
  select(id, distance)

items <- c("EDE_A", "EDE_B", "EDE_C", "EDE_D", "EDE_E")

# Create all unique pairs (order does not matter)
pair_list_db <- combn(items, 2) %>%
  # Transpose the matrix so each row is a pair of items
  t() %>%
  # Convert the matrix into a tibble
  as_tibble() %>%
  # Rename the columns from V1 and V2 to 'item1' and 'item2'
  rename(item1 = V1, item2 = V2)
# Function to calculate group-significant edges
# where the selected distance is shorter than all other distances
calculate_significance <- function(var1, var2, data) {
  data %>%
    # Create grouping variable: group = FALSE for var1-var2 pairs
    mutate(group = !((Var1 == var1 & Var2 == var2) | (Var1 == var2 & Var2 == var1))) %>%
    # Fit a linear model: predict distance (value) by group
    lm(value ~ group, data = .) %>%
    # Extract 'group' variable  results (coefficient table)
    broom::tidy() %>% slice(2) %>%
    # Check if p-value is significant (p < 0.05) and estimate is positive
    summarise(distance = if_else(p.value < 0.05 & estimate > 0, estimate, 0)) %>%
    # Pull out the distance
    pull(distance)
}

# Prepare the data for group comparison
undirected2 <- undirected1 %>%
  pull(distance) %>%
  # Turn distances into long format
  map(melt) %>%
  # Combine all melted matrices into one big tibble
  bind_rows()

# Calculate group-significant edges (where distances are shorter than all other distances)
undirected3 <- map2_dbl(pair_list_db$item1, pair_list_db$item2, calculate_significance,
undirected2) %>%
  # Combine the calculated distances with the corresponding item pairs
  cbind(pair_list_db) %>%
  as_tibble() %>%
  select(item1, item2, value = 1)

# Make symmetrical distance matrix for plotting
undirected4 <- undirected3 %>%
  bind_rows(undirected3 %>%
              rename(item2 = item1, item1 = item2)) %>%
```

```
  # Cast into a wide matrix format: rows = item1, columns = item2
  acast(item1 ~ item2, value.var = "value")

# --------------------------
# Visualize the group-level undirected network
# --------------------------

qgraph(undirected4, layout = "spring")
```