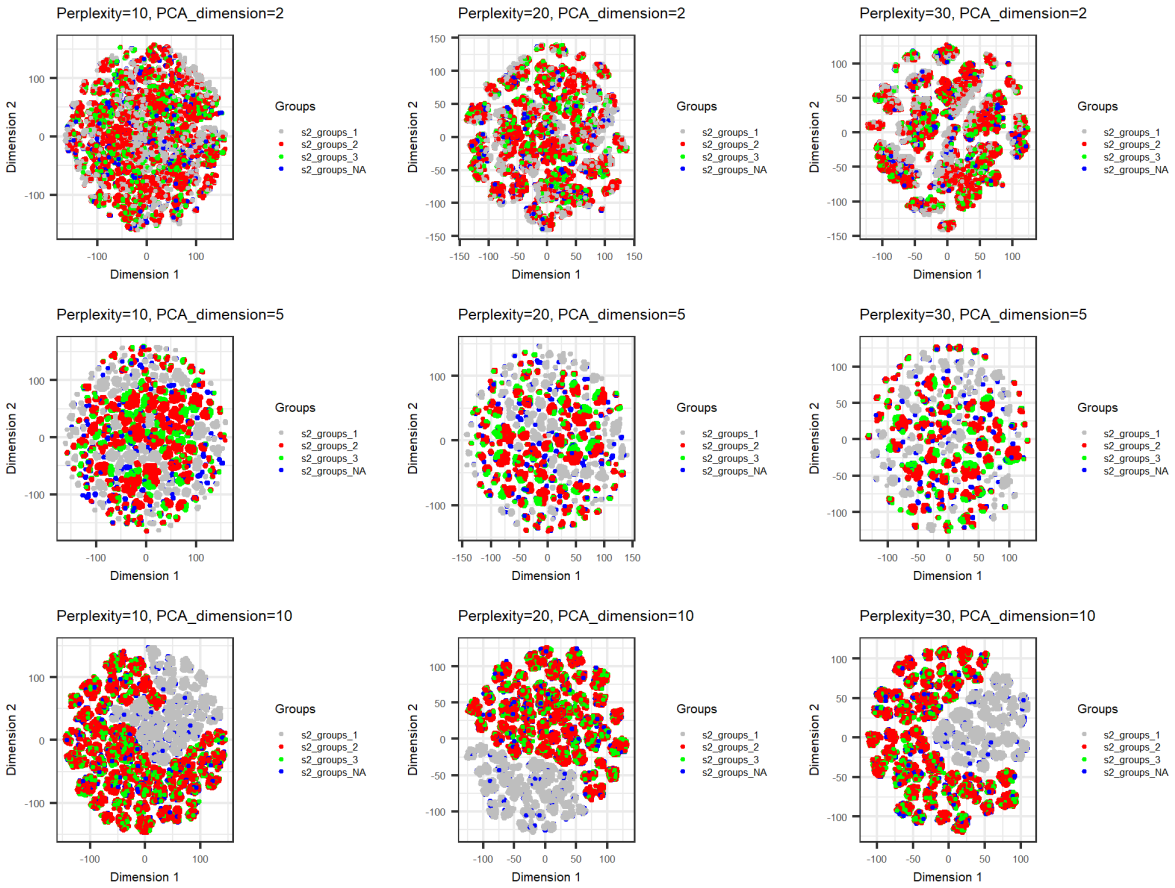
Running tSNE on UKB2 with a longer max_iter: 2500

```
# Performing t-SNE
# Set seed for reproducibility
set.seed(42)

# Define parameters
max_iter <- 2500 # Updated maximum iterations
theta <- 0.1
perplexities <- c(10, 20, 30) # Updated perplexity values
pcaDims <- c(2, 5, 10)
mycolors <- c("s2_groups_1" = "gray", "s2_groups_2" = "red",
              "s2_groups_3" = "green", "s2_groups_NA" = "blue")
figWidth <- 2000
pointSize <- 0.5
legendSize <- 5
textSize <- 5
num_threads <- 0

# Function to perform t-SNE with PCA initialization and create plots
doRtsne <- function(perplexity, pcaDim) {
  tsne <- Rtsne(data_final[, !(names(data_final) %in% c("FID_71392",
                                                        "FID_71392.1",
                                                        "s2_groups_1",
                                                        "s2_groups_2",
                                                        "s2_groups_3",
                                                        "s2_groups_NA"))],
                initial_dims = pcaDim,
                dims = 2,
                perplexity = perplexity,
                verbose = TRUE,
                max_iter = max_iter,
                theta = theta,
                num_threads = num_threads)

  # Recreate the Groups column for plotting
```

```
  tsne_plot <- data.frame(x = tsne$Y[, 1], y = tsne$Y[, 2], Groups =
                            factor(
                              apply(data_final[, c("s2_groups_1",
"s2_groups_2",
                                                   "s2_groups_3",
"s2_groups_NA")], 1, function(x) {
                                                     if
(!is.na(x["s2_groups_1"]) && x["s2_groups_1"] == 1)
return("s2_groups_1")
                                                     if
(!is.na(x["s2_groups_2"]) && x["s2_groups_2"] == 1)
return("s2_groups_2")
                                                     if
(!is.na(x["s2_groups_3"]) && x["s2_groups_3"] == 1)
return("s2_groups_3")
                                                     if
(!is.na(x["s2_groups_NA"]) && x["s2_groups_NA"] == 1)
return("s2_groups_NA")
                                                     return("Unknown") #
Return "Unknown" if no group matches
                                                   })
                            ))

  plot <- ggplot(tsne_plot) +
    geom_point(aes(x = x, y = y, color = Groups), size = pointSize) +
    scale_color_manual(values = mycolors) +
    ggtitle(paste0("Perplexity=", perplexity, ", PCA_dimension=", pcaDim))
+
    xlab("Dimension 1") +
    ylab("Dimension 2") +
    theme_bw() +
    theme(text = element_text(size = textSize),
          legend.key.size = unit(legendSize, "point"))

  return(plot)
}
```

```
# Define your datasets (if you only have one dataset `data_final`, just use
it directly)
datasets <- list(
  data_final = data_final
)


# Iterate over datasets
for (datname in names(datasets)) {
  dat <- datasets[[datname]]


  # Check for required grouping columns
  if (all(c("s2_groups_1", "s2_groups_2", "s2_groups_3",
            "s2_groups_NA") %in% names(dat))) {


    # Initialize a list to store plots
    pls <- list()


    # Perform t-SNE and plot for each PCA dimension and perplexity
combination
    for (pcaDim in pcaDims) {
      plots <- lapply(perplexities, function(perplexity)
        doRtsne(perplexity, pcaDim))
      pls <- c(pls, plots)
    }


    # Arrange plots in a grid
    grid_plot_filename <- paste0("tsne_2d_grid_", datname, ".png")
    png(grid_plot_filename, width = figWidth, height = figWidth * 0.75,
units = "px", res = 300)
    grid.arrange(grobs = pls, nrow = length(pcaDims), ncol =
length(perplexities))
    dev.off()


  } else {
    warning(paste("Some one-hot encoded columns are missing in", datname))
  }
}
```



The output is fairly the same as before when max_iter was set to 1000.

Trying out some new PCA Dimensions:

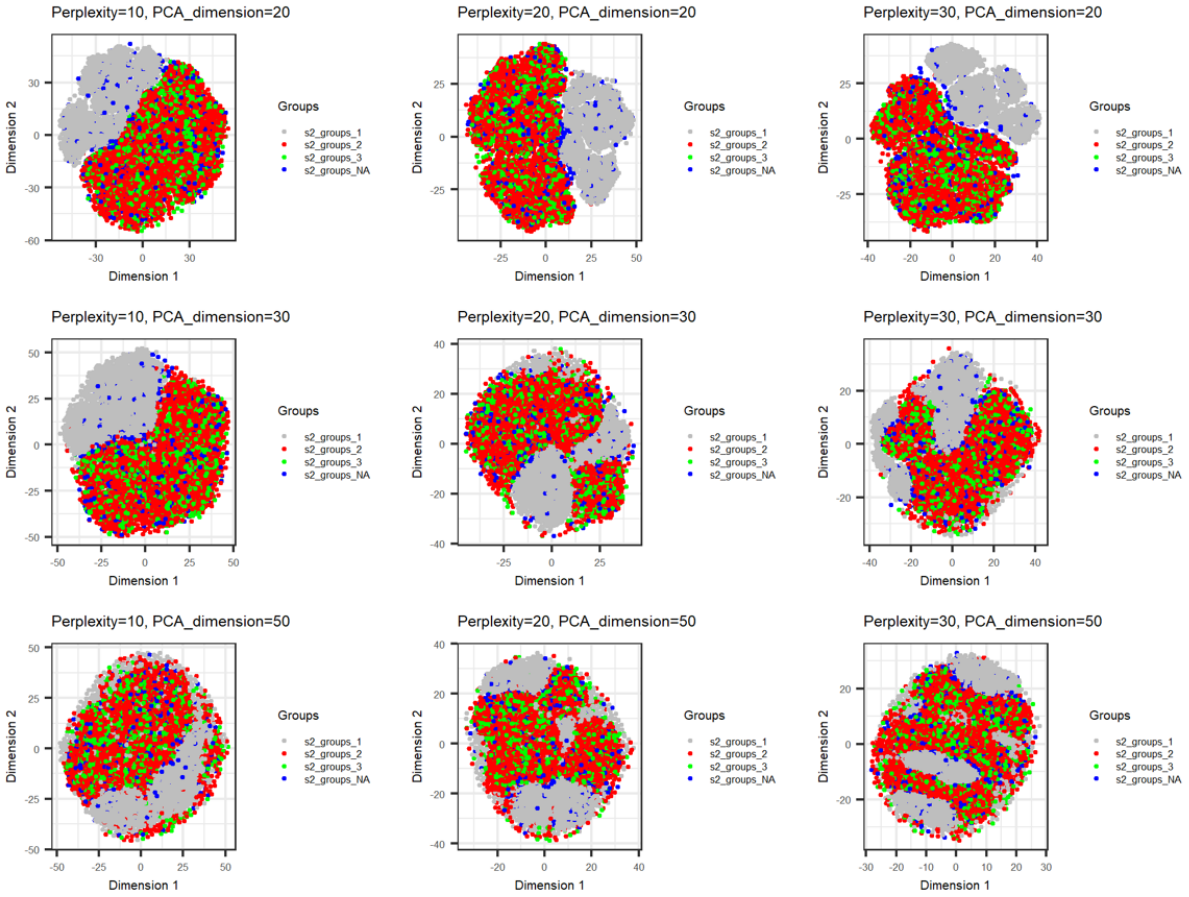
Reason – Bengt's comment:

- *pcaDims*; The number of principal components to input to t_SNE. This can also affect the results fairly much. I've been recommended taht we should probably keep this rather low since our PCA plot indicate that we have much info in the first components. I try three different *pcaDim* values (*pcaDim* is my abbreviation – not standard:) *This is not used for the distance matrix indata*.

How to choose a new dimension:

Consideration	Recommended PCA Dimensions
Data complexity	30-50 dimensions
Data size	Larger datasets: 50-100 dimensions
Quality vs Speed	Higher dimensions improve quality but increase the computational cost; 30 dimensions is a good balance

Using PCA dimensions 20, 30, 50:



Following this, I am attempting to visualise in 3D to see maybe spatial differences will exist between groups 2 and 3, and maybe we will see some differences between the groups since they are intertwined in the 2D visual.

```
##TRYING OUT 3D


install.packages("plotly")
library(plotly)
# Set seed for reproducibility
set.seed(42)
# Define parameters
max_iter <- 1000 # Reduced maximum iterations
theta <- 0.1
perplexities <- c(10, 20) # Reduced perplexity values for faster
computation
pcaDims <- c(20, 30) # Reduced PCA dimensions for faster computation
mycolors <- c("s2_groups_1" = "gray", "s2_groups_2" = "red",
              "s2_groups_3" = "green", "s2_groups_NA" = "blue")
figWidth <- 2000
pointSize <- 0.5
legendSize <- 5
textSize <- 5
num_threads <- parallel::detectCores() # Use all available cores

# Function to perform t-SNE with PCA initialization and create 3D plots
doRtsne <- function(perplexity, pcaDim) {
  tsne <- Rtsne(data_final[, !(names(data_final) %in% c("FID_71392",
                                                        "FID_71392.1",
                                                        "s2_groups_1",
                                                        "s2_groups_2",
                                                        "s2_groups_3",
                                                        "s2_groups_NA"))],
                initial_dims = pcaDim,
                dims = 3, # Changed to 3 dimensions
                perplexity = perplexity,
                verbose = TRUE,
                max_iter = max_iter,
```

```
                theta = theta,
                num_threads = num_threads)


  # Recreate the Groups column for plotting
  tsne_plot <- data.frame(x = tsne$Y[, 1], y = tsne$Y[, 2], z = tsne$Y[,
3], Groups =
                            factor(
                              apply(data_final[, c("s2_groups_1",
"s2_groups_2",
                                                   "s2_groups_3",
"s2_groups_NA")], 1, function(x) {
                                                     if
(!is.na(x["s2_groups_1"]) && x["s2_groups_1"] == 1)

return("s2_groups_1")
                                                     if
(!is.na(x["s2_groups_2"]) && x["s2_groups_2"] == 1)

return("s2_groups_2")
                                                     if
(!is.na(x["s2_groups_3"]) && x["s2_groups_3"] == 1)

return("s2_groups_3")
                                                     if
(!is.na(x["s2_groups_NA"]) && x["s2_groups_NA"] == 1)

return("s2_groups_NA")
                                                     return("Unknown") #
Return "Unknown" if no group matches
                                                   })
                            ))


  plot <- plot_ly(tsne_plot, x = ~x, y = ~y, z = ~z, color = ~Groups,
colors = mycolors, type = 'scatter3d', mode = 'markers') %>%
    layout(title = paste0("Perplexity=", perplexity, ", PCA_dimension=",
pcaDim),
           scene = list(xaxis = list(title = 'Dimension 1'),
                        yaxis = list(title = 'Dimension 2'),
                        zaxis = list(title = 'Dimension 3')))


  return(plot)
}
```

```
# Define your datasets (if you only have one dataset `data_final`, just use
it directly)
datasets <- list(
  data_final = data_final
)


# Iterate over datasets
for (datname in names(datasets)) {
  dat <- datasets[[datname]]


  # Check for required grouping columns
  if (all(c("s2_groups_1", "s2_groups_2", "s2_groups_3",
            "s2_groups_NA") %in% names(dat))) {


    # Initialize a list to store plots
    pls <- list()


    # Perform t-SNE and plot for each PCA dimension and perplexity
combination
    for (pcaDim in pcaDims) {
      plots <- lapply(perplexities, function(perplexity)
        doRtsne(perplexity, pcaDim))
      pls <- c(pls, plots)
    }


    # Save plots as HTML files (as Plotly is interactive)
    for (i in seq_along(pls)) {
      plot_filename <- paste0("tsne_3d_perplexity_", perplexities[i],
"_pcaDim_", pcaDims[ceiling(i / length(perplexities))], ".html")
      htmlwidgets::saveWidget(pls[[i]], file = plot_filename)
    }


  } else {
    warning(paste("Some one-hot encoded columns are missing in", datname))
  }
}
```

1. https://gla-my.sharepoint.com/personal/2945425o_student_gla_ac_uk/Documents/Documents/HDR%20UK%20Internship%20Project/Trial%20Week%20-%20Test%20Runs/tsne_3d_perplexity_10_pcaDim_20.html
2. https://gla-my.sharepoint.com/personal/2945425o_student_gla_ac_uk/Documents/Documents/HDR%20UK%20Internship%20Project/Trial%20Week%20-%20Test%20Runs/tsne_3d_perplexity_20_pcaDim_20.html
3. https://gla-my.sharepoint.com/personal/2945425o_student_gla_ac_uk/Documents/Documents/HDR%20UK%20Internship%20Project/Trial%20Week%20-%20Test%20Runs/tsne_3d_perplexity_NA_pcaDim_30.html