

```

# =====
# TITLE: Metagenomic EDA Pipeline: Interactive Diversity & Composition
# DESCRIPTION: A Phyloseq-based workflow for cleaning, analyzing, and
#               visualizing microbiome census data.
# =====

# -----
# SECTION 1: SETUP & CONFIGURATION
# -----
output_dir <- "D:/DOWNLOADS"
if (!dir.exists(output_dir)) dir.create(output_dir, recursive = TRUE)

# Load Libraries
library(phyloseq)
library(ggplot2)
library(vegan)
library(plotly)
library(DT)
library(htmlwidgets)

# -----
# SECTION 2: DATA LOADING & SPEED OPTIMIZATION
# -----
message("Step 2: Loading & Optimizing Data...")

data("GlobalPatterns")
physeq <- GlobalPatterns

# --- OPTIMIZATION FOR SPEED ---
# Keep only the TOP 2,000 most abundant taxa (out of ~19,000).
# This prevents the script from freezing during plotting.
top_taxa <- names(sort(taxa_sums(physeq), decreasing = TRUE)[1:2000])
physeq_filt <- prune_taxa(top_taxa, physeq)

# Normalize to relative abundance
physeq_rel <- transform_sample_counts(physeq_filt, function(x) x / sum(x))
message("    [OK] Data filtered to top 2,000 taxa.")

# -----
# SECTION 3: RAREFACTION CURVE (FAST & FIXED)
# -----
message("Step 3: Saving Rarefaction Curve...")

png(filename = file.path(output_dir, "01_Rarefaction_Curve.png"), width = 800,
height = 600)

# 1. Select first 10 samples for the curve (Sufficient for QC)
rare_data <- prune_samples(sample_names(physeq_filt)[1:10], physeq_filt)

# 2. Fix "logical subscript" error by converting to matrix
rare_mat <- as(otu_table(rare_data), "matrix")
if(taxa_are_rows(rare_data)) { rare_mat <- t(rare_mat) }

# 3. Run with large step size (500) for speed
rarecurve(rare_mat, step = 500, col = "blue", label = FALSE,
           main = "Rarefaction Curve (Fast Preview - Top Taxa)",
           ylab = "OTUs Detected", xlab = "Reads Sequenced")
dev.off()
message("    [OK] Rarefaction Curve Saved.")

# -----
# SECTION 4: ALPHA DIVERSITY
# -----
message("Step 4: Saving Alpha Diversity...")

```

```

p_alpha <- plot_richness(physeq_filt, x = "SampleType", measures = c("Shannon",
"Observed")) +
  geom_boxplot(aes(fill = SampleType), alpha = 0.6) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ggtitle("Alpha Diversity (Top 2000 Taxa)")

ggsave(file.path(output_dir, "02_Alpha_Diversity.png"), plot = p_alpha, width =
10, height = 8)
saveWidget(ggplotly(p_alpha), file.path(output_dir,
"02_Alpha_Diversity_Interactive.html"))

# -----
# SECTION 5: BETA DIVERSITY (PCoA)
# -----
message("Step 5: Saving PCoA...")

ord <- ordinate(physeq_rel, method = "PCoA", distance = "bray")

p_beta <- plot_ordination(physeq_rel, ord, color = "SampleType") +
  geom_point(size = 3, alpha = 0.7) +
  stat_ellipse(type = "norm", linetype = 2) +
  ggtitle("PCoA - Bray-Curtis Dissimilarity") +
  theme_bw()

ggsave(file.path(output_dir, "03_Beta_Diversity_PCoA.png"), plot = p_beta, width
= 10, height = 8)
saveWidget(ggplotly(p_beta), file.path(output_dir,
"03_Beta_Diversity_Interactive.html"))

# -----
# SECTION 6: COMPOSITION BARPLOT
# -----
message("Step 6: Saving Bar Plot...")

# Top 10 Phyla within the filtered dataset
top10 <- names(sort(taxa_sums(physeq_rel), decreasing = TRUE)[1:10])
physeq_phylum_top10 <- prune_taxa(top10, physeq_rel)

p_bar <- plot_bar(physeq_phylum_top10, x = "SampleType", fill = "Phylum") +
  geom_bar(stat = "identity", position = "stack") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ggtitle("Top 10 Phyla Composition")

ggsave(file.path(output_dir, "04_Taxonomy_Barplot.png"), plot = p_bar, width =
12, height = 8)
saveWidget(ggplotly(p_bar), file.path(output_dir,
"04_Taxonomy_Barplot_Interactive.html"))

# -----
# SECTION 7: HEATMAP
# -----
message("Step 7: Saving Heatmap...")

# Top 20 Genera within the filtered dataset
top20 <- names(sort(taxa_sums(physeq_rel), decreasing = TRUE)[1:20])
physeq_top20 <- prune_taxa(top20, physeq_rel)

p_heat <- plot_heatmap(physeq_top20, taxa.label = "Genus", sample.label =
"SampleType",
                       low = "white", high = "darkblue", na.value = "white") +
  ggtitle("Heatmap of Top 20 Genera")

ggsave(file.path(output_dir, "05_Heatmap.png"), plot = p_heat, width = 12,

```

```
height = 8)
saveWidget(ggplotly(p_heat), file.path(output_dir,
"05_Heatmap_Interactive.html"))

# -----
# SECTION 8: EXPORT TABLES
# -----
message("Step 8: Exporting Data Tables...")

write.csv(as.data.frame(otu_table(physeq_filt)), file.path(output_dir,
"06_GlobalPatterns_0TU.csv"))
tax_df <- as.data.frame(tax_table(physeq_filt))
saveWidget(datatable(tax_df, caption = "Taxonomy", filter = 'top'),
file.path(output_dir, "06_Taxonomy_Table.html"))

# -----
# COMPLETION
# -----
message("====")
message("ANALYSIS COMPLETE")
message("All files saved to: ", output_dir)
message("====")
```