exploratory.R

```r
#####################################################################
# 0. Carregar subset das máquinas
#####################################################################

load_machines_subset <- function() {
  library(rrcov)
  data("machines")

  start <- which(rownames(machines) == "hp-3000/64")
  end   <- which(rownames(machines) == "ibm-4331-2")

  subset <- machines[start:end, ]
  return(subset)
}

subset <- load_machines_subset()

#####################################################################
# 1. Exploratory Data Analysis (EDA)
#####################################################################

library(ggplot2)
library(GGally)
library(psych)
library(corrplot)

# Remover os rownames para facilitar gráficos
df <- subset
df$Machine <- rownames(df)
rownames(df) <- NULL

#####################################################################
# 1.1 Estatísticas descritivas
#####################################################################

# Summary
summary(df)

# Média
means <- colMeans(df[ , -ncol(df)])
means

# Mediana
medians <- apply(df[ , -ncol(df)], 2, median)
medians

# Trimmed mean (10%)
trimmed_means <- apply(df[ , -ncol(df)], 2, mean, trim = 0.1)
trimmed_means

# Winsorized mean (10%)
winsor_means <- apply(df[ , -ncol(df)], 2, winsor.mean)
winsor_means

# Variâncias
variances <- apply(df[ , -ncol(df)], 2, var)
```

exploratory.R

```r
variances

# MAD (Median Absolute Deviation)
mads <- apply(df[ , -ncol(df)], 2, mad)
mads

#####################################################################
# 1.2 Covariância, Variância Total, Generalized Variance
#####################################################################

S <- cov(df[, -ncol(df)])  # matriz de covariância

# Variância Total = soma das variâncias
total_variance <- sum(diag(S))
total_variance

# Generalized Variance (determinante)
generalized_variance <- det(S)
generalized_variance

#####################################################################
# 1.3 Distâncias de Mahalanobis
#####################################################################

center <- colMeans(df[ , -ncol(df)])
md <- mahalanobis(df[ , -ncol(df)], center, S)

df$Mahalanobis <- md
md

# Outliers potenciais (nível 97.5%)
cutoff <- qchisq(0.975, df = ncol(df)-1)
which(md > cutoff)

#####################################################################
# 1.4 Gráficos
#####################################################################

# Histogramas
par(mfrow=c(3,3))
for(col in names(df)[-c(ncol(df), length(names(df)))]){
  hist(df[[col]], main=paste("Histograma de", col), xlab=col)
}

# Boxplots
par(mfrow=c(3,3))
for(col in names(df)[-c(ncol(df), length(names(df)))]){
  boxplot(df[[col]], main=paste("Boxplot de", col))
}

# Scatterplot matrix
GGally::ggpairs(df[, sapply(df, is.numeric)])

# Correlograma
corrplot(cor(df[sapply(df, is.numeric)]),
         method = "color", addCoef.col = "black")
```

exploratory.R

load_machines_subset.R

```
load_machines_subset <- function() {
  if (!requireNamespace("rrcov", quietly = TRUE)) {
    install.packages("rrcov")
  }
  library(rrcov)
  data("machines")
  dim(machines)

  i1 <- which(rownames(machines) == "hp-3000/64")
  i2 <- which(rownames(machines) == "ibm-4331-2")



  new <- machines[i1:i2, ]
  new
}
```

pca_classic_outlier.R

```r
source("scripts/load_machines_subset.R")
X <- load_machines_subset()

#introduzir outlier
i1 <- which(rownames(X) == "hp-3000/64")
X[i1, ] <- c(75, 2000, 0.8, 80000, 300, 24, 62, 47)


source("scripts/pca_original_scale.R")
compute_pca(X, "plots/plots_pca_original_outlier", std = FALSE)
```

pca_original_scale.R

```r
source("scripts/load_machines_subset.R")

compute_pca <- function(X = load_machines_subset(),
                                out_dir = "plots/plots_pca_original", std = FALSE) {

  pca <- prcomp(X, center = TRUE, scale. = std)
  pve  <- (pca$sdev^2) / sum(pca$sdev^2)
  cpve <- cumsum(pve)
  k <- which(cpve >= 0.95)[1]

  cat("\n==================== RESULTS ====================\n")
  cat("Number of variables (p):", ncol(X), "\n")
  cat("Number of observations (n):", nrow(X), "\n\n")

  cat("PVE (proportion variance explained) per PC:\n")
  print(pve)

  cat("\nCPVE (cumulative PVE):\n")
  print(cpve)

  cat("\nMinimum k for CPVE >= 0.95:\n")
  cat("k =", k, "\n")
  cat("CPVE[k] =", cpve[k], "\n")


  cat("Loadings (rotation) for retained PCs (1..k):\n")
  print(pca$rotation[, 1:k, drop = FALSE])

  abs_load <- abs(pca$rotation)
  top_pc1 <- sort(abs_load[, 1], decreasing = TRUE)
  cat("\nTop contributors to PC1 (absolute loading):\n")
  print(top_pc1)

  if (ncol(X) >= 2) {
    top_pc2 <- sort(abs_load[, 2], decreasing = TRUE)
    cat("\nTop contributors to PC2 (absolute loading):\n")
    print(top_pc2)
  }


  if (!dir.exists(out_dir))
    dir.create(out_dir)

  # ------------------------------
  # PLOT 1: Scree plot (PVE)
  # ------------------------------
  png(
    filename = file.path(out_dir, "01_scree_pve.png"),
    width = 1200,
    height = 800,
    res = 150
  )
  barplot(
    pve,
    names.arg = paste0("PC", seq_along(pve)),
    las = 2,
```

pca_original_scale.R

```
  xlab = "Principal components",
  ylab = "Proportion of variance explained"
)
dev.off()

# ------------------------------
# PLOT 2: Cumulative variance + 95%
# ------------------------------
png(
  filename = file.path(out_dir, "02_cumulative_cpve.png"),
  width = 1200,
  height = 800,
  res = 150
)
plot(
  cpve,
  type = "b",
  pch = 19,
  ylim = c(0, 1),
  xlab = "Number of PCs",
  ylab = "Cumulative proportion of variance explained"
)
abline(h = 0.95, lty = 2)
abline(v = k, lty = 3)
text(
  k,
  cpve[k],
  labels = paste0("k = ", k, " (", round(100 * cpve[k], 2), "%)"),
  pos = 4,
  cex = 0.9
)
dev.off()

# ------------------------------
# PLOT 3: Scores plot (PC1 vs PC2)
# ------------------------------

library(ggplot2)
library(ggrepel)

scores <- pca$x
rownames(scores) <- rownames(X)

scores_df <- data.frame(
  PC1  = scores[, 1],
  PC2  = scores[, 2],
  name = rownames(scores)
)

med1 <- median(scores_df$PC1)
med2 <- median(scores_df$PC2)

scores_df$dist <- sqrt((scores_df$PC1 - med1)^2 + (scores_df$PC2 - med2)^2)

# top 25% have label
thr <- quantile(scores_df$dist, 0.75)
```

pca_original_scale.R

```
  scores_df$label <- ifelse(scores_df$dist > thr, scores_df$name, "")

  # jitter to split coincident points
  scores_df$PC1_j <- jitter(scores_df$PC1, amount = diff(range(scores_df$PC1)) * 0.01
)
  scores_df$PC2_j <- jitter(scores_df$PC2, amount = diff(range(scores_df$PC2)) * 0.01
)

  png(
    filename = file.path(out_dir, "03_scores_pc1_pc2.png"),
    width  = 800,
    height = 800,
    res    = 150
  )

  p <- ggplot(scores_df, aes(PC1_j, PC2_j)) +
    geom_point() +
    geom_vline(xintercept = 0, linetype = 2) +
    geom_hline(yintercept = 0, linetype = 2) +
    geom_text_repel(
      aes(label = label),
      size        = 3.5,
      max.overlaps = 50
    ) +
    labs(
      title = "Scores plot (PC1 vs PC2)",
      x     = "PC1 score",
      y     = "PC2 score"
    ) +
    coord_fixed() +
    theme_classic(base_size = 14) +
    theme(aspect.ratio = 1)

  print(p)
  dev.off()

  # --------------------------------
  # PLOT 4: Loadings barplots (PC1 and PC2)
  # --------------------------------
  png(
    filename = file.path(out_dir, "04_loadings_pc1_pc2.png"),
    width = 1400,
    height = 800,
    res = 150
  )
  par(mfrow = c(1, 2), mar = c(7, 4, 4, 1))

  barplot(pca$rotation[, 1],
          las = 2,
          main = "Loadings - PC1",
          ylab = "Loading")
  abline(h = 0, lty = 2)

  barplot(pca$rotation[, 2],
          las = 2,
          main = "Loadings - PC2",
```

pca_original_scale.R

```r
        ylab = "Loading")
abline(h = 0, lty = 2)

par(mfrow = c(1, 1))
dev.off()

# ------------------------------
# PLOT 5: Biplot (PC1 vs PC2)
# ------------------------------
png(
  filename = file.path(out_dir, "05_biplot_pc1_pc2.png"),
  width = 1200,
  height = 900,
  res = 150
)
biplot(pca,
      choices = c(1, 2),
      cex = 0.8,
      main = "")
title("Biplot (PC1 vs PC2) - scores + loadings", line = 2)
dev.off()

# ======================================================================
# Extreme analysis based on loadings + var values
# ======================================================================

extremes <- scores_df[order(-scores_df$dist), ]
ext_names <- extremes$name[1:6]
load <- pca$rotation

# most contribution for pc1 and pc2
vars_pc1 <- names(sort(abs(load[, 1]), decreasing = TRUE))[1:5]
vars_pc2 <- names(sort(abs(load[, 2]), decreasing = TRUE))[1:5]
vars_key <- unique(c(vars_pc1, vars_pc2))
X_ext <- X[ext_names, vars_key, drop = FALSE]

cat("\nScores of the machines farthest in the PC1–PC2 space:\n")
print(head(extremos, 6))

cat("\nMachines farthest in the PC1–PC2 space:\n")
print(ext_names)

cat("\nVariables with the highest loadings on PC1:\n")
print(vars_pc1)

cat("\nVariables with the highest loadings on PC2:\n")
print(vars_pc2)

cat("\nOriginal values of these variables for each extreme machine:\n")
print(round(X_ext, 3))

# ======================================================================


cat("Saved plots in folder:", out_dir, "\n")
cat("Files:\n")
```

pca_original_scale.R

```
  cat("  01_scree_pve.png\n")
  cat("  02_cumulative_cpve.png\n")
  cat("  03_scores_pc1_pc2.png\n")
  cat("  04_loadings_pc1_pc2.png\n")
  cat("  05_biplot_pc1_pc2.png\n")
}

compute_pca()
```

robust_pca.R

```r
source("scripts/load_machines_subset.R")

perform_robust_pca <- function(alpha = 0.75,
                               out_dir = "plots/plots_pca_robust_mcd") {
  needed <- c("rrcov", "ggplot2", "ggrepel")
  missing_pkgs <- needed[!vapply(needed, requireNamespace, logical(1), quietly = TRUE
)]
  if (length(missing_pkgs) > 0) {
    install.packages(missing_pkgs)
  }
  lapply(needed, library, character.only = TRUE)

  X <- load_machines_subset()
  outlier_name <- "hp-3000/64"

  # introduce same gross outlier used in the classical PCA experiment
  i1 <- which(rownames(X) == outlier_name)
  if (length(i1) == 1) {
    X[i1, ] <- c(75, 2000, 0.8, 80000, 300, 24, 62, 47)
  } else {
    warning("hp-3000/64 not found; proceeding without injected outlier.")
    outlier_name <- NA
  }

  if (!dir.exists(out_dir)) {
    dir.create(out_dir, recursive = TRUE)
  }

  cov_mcd <- CovMcd(X, alpha = alpha)
  eig <- eigen(cov_mcd@cov)
  eig_values <- eig$values
  eig_values[eig_values < 0] <- 0

  loadings <- eig$vectors
  colnames(loadings) <- paste0("PC", seq_len(ncol(loadings)))
  rownames(loadings) <- colnames(X)

  sdev <- sqrt(eig_values)
  pve <- eig_values / sum(eig_values)
  cpve <- cumsum(pve)
  k <- which(cpve >= 0.95)[1]

  centered <- sweep(as.matrix(X), 2, cov_mcd@center, "-")
  scores <- centered %*% loadings
  rownames(scores) <- rownames(X)
  colnames(scores) <- paste0("PC", seq_len(ncol(scores)))

  weights <- ifelse(cov_mcd@wt == 1, "MCD set", "Flagged")
  weights <- factor(weights, levels = c("MCD set", "Flagged"))

  cat("\n================ ROBUST PCA (MCD) ================\n")
  cat("Alpha (subset proportion):", alpha, "\n")
  cat("Number of variables (p):", ncol(X), "\n")
  cat("Number of observations (n):", nrow(X), "\n\n")

  cat("PVE per PC (robust variance ratios):\n")
```

robust_pca.R

```
  print(round(pve, 4))

  cat("\nCPVE (cumulative PVE):\n")
  print(round(cpve, 4))

  cat("\nMinimum k for CPVE >= 0.95:\n")
  cat("k =", k, "\n")
  cat("CPVE[k] =", round(cpve[k], 4), "\n\n")

  cat("Loadings for PCs 1..k:\n")
  print(round(loadings[, 1:k, drop = FALSE], 4))

  abs_load <- abs(loadings)
  top_pc1 <- sort(abs_load[, 1], decreasing = TRUE)
  cat("\nTop contributors to PC1 (absolute loading):\n")
  print(round(top_pc1, 4))

  if (ncol(X) >= 2) {
    top_pc2 <- sort(abs_load[, 2], decreasing = TRUE)
    cat("\nTop contributors to PC2 (absolute loading):\n")
    print(round(top_pc2, 4))
  }

  scores_df <- data.frame(
    PC1 = scores[, 1],
    PC2 = scores[, 2],
    name = rownames(scores),
    group = weights,
    stringsAsFactors = FALSE
  )

  med1 <- median(scores_df$PC1)
  med2 <- median(scores_df$PC2)
  scores_df$dist <- sqrt((scores_df$PC1 - med1)^2 + (scores_df$PC2 - med2)^2)
  scores_df$label <- ifelse(!is.na(outlier_name) & scores_df$name == outlier_name,
                            scores_df$name, "")
  scores_df$PC1_j <- jitter(scores_df$PC1, amount = diff(range(scores_df$PC1)) * 0.01
)
  scores_df$PC2_j <- jitter(scores_df$PC2, amount = diff(range(scores_df$PC2)) * 0.01
)
  scores_df$highlight <- ifelse(!is.na(outlier_name) & scores_df$name == outlier_name
,
                                "Injected outlier", "Other machines")

  # Scree plot (PVE)
  png(
    filename = file.path(out_dir, "01_scree_pve.png"),
    width = 1200,
    height = 800,
    res = 150
  )
  barplot(
    pve,
    names.arg = paste0("PC", seq_along(pve)),
    las = 2,
    xlab = "Principal components",
```

robust_pca.R

```
  ylab = "Proportion of variance explained"
)
dev.off()

# Cumulative variance
png(
  filename = file.path(out_dir, "02_cumulative_cpve.png"),
  width = 1200,
  height = 800,
  res = 150
)
plot(
  cpve,
  type = "b",
  pch = 19,
  ylim = c(0, 1),
  xlab = "Number of PCs",
  ylab = "Cumulative proportion of variance explained"
)
abline(h = 0.95, lty = 2)
abline(v = k, lty = 3)
text(k,
     cpve[k],
     labels = paste0("k = ", k, " (", round(100 * cpve[k], 2), "%)"),
     pos = 4)
dev.off()

# Scores PC1 vs PC2
png(
  filename = file.path(out_dir, "03_scores_pc1_pc2.png"),
  width = 800,
  height = 800,
  res = 150
)
p <- ggplot(scores_df, aes(PC1_j, PC2_j, color = highlight, shape = group)) +
  geom_point(size = 2.8) +
  geom_vline(xintercept = 0, linetype = 2) +
  geom_hline(yintercept = 0, linetype = 2) +
  geom_text_repel(
    aes(label = label),
    size = 3.5,
    max.overlaps = 50
  ) +
  scale_color_manual(values = c("Injected outlier" = "#d73027",
                                "Other machines" = "#1b9e77")) +
  scale_shape_manual(values = c("MCD set" = 16, "Flagged" = 17)) +
  labs(
    title = "Robust PCA scores (PC1 vs PC2)",
    x = "PC1 score",
    y = "PC2 score",
    color = "",
    shape = "MCD weight"
  ) +
  coord_fixed() +
  theme_classic(base_size = 14) +
  theme(aspect.ratio = 1)
```

robust_pca.R

```r
print(p)
dev.off()

png(
  filename = file.path(out_dir, "04_loadings_pc1_pc2.png"),
  width = 1400,
  height = 800,
  res = 150
)
par(mfrow = c(1, 2), mar = c(7, 4, 4, 1))
barplot(loadings[, 1],
        las = 2,
        main = "Robust loadings - PC1",
        ylab = "Loading")
abline(h = 0, lty = 2)

barplot(loadings[, 2],
        las = 2,
        main = "Robust loadings - PC2",
        ylab = "Loading")
abline(h = 0, lty = 2)
par(mfrow = c(1, 1))
dev.off()

prcomp_like <- list(
  sdev = sdev,
  rotation = loadings,
  center = cov_mcd@center,
  scale = FALSE,
  x = scores
)
class(prcomp_like) <- "prcomp"
png(
  filename = file.path(out_dir, "05_biplot_pc1_pc2.png"),
  width = 1200,
  height = 900,
  res = 150
)
biplot(prcomp_like,
       choices = c(1, 2),
       cex = 0.8,
       main = "")
title("Robust biplot (PC1 vs PC2)", line = 2)
dev.off()

# Outlier map: score distance vs orthogonal distance
q <- if (is.na(k)) min(ncol(X), 2) else max(1, min(k, ncol(X)))
score_subset <- scores[, 1:q, drop = FALSE]
lambda_subset <- eig_values[1:q]
lambda_subset[lambda_subset <= .Machine$double.eps] <- .Machine$double.eps
score_dist <- sqrt(rowSums((score_subset^2) / matrix(lambda_subset,
                                                     nrow = nrow(score_subset),
                                                     ncol = q,
                                                     byrow = TRUE)))
recon <- score_subset %*% t(loadings[, 1:q, drop = FALSE])
X_hat <- sweep(recon, 2, cov_mcd@center, "+")
```

robust_pca.R

```
  residuals <- as.matrix(X) - X_hat
  orth_dist <- sqrt(rowSums(residuals^2))
  sd_thresh <- sqrt(qchisq(0.975, df = q))
  od_thresh <- median(orth_dist) + mad(orth_dist) * sqrt(qchisq(0.975, df = max(1, nc
ol(X) - q)))

  outlier_map_df <- data.frame(
    name = rownames(X),
    ScoreDistance = score_dist,
    OrthDistance = orth_dist,
    group = weights,
    highlight = ifelse(!is.na(outlier_name) & rownames(X) == outlier_name,
                       "Injected outlier", "Other machines")
  )

  png(
    filename = file.path(out_dir, "06_outlier_map.png"),
    width = 1000,
    height = 750,
    res = 150
  )
  p_map <- ggplot(outlier_map_df,
                  aes(ScoreDistance, OrthDistance,
                      color = highlight, shape = group)) +
    geom_point(size = 3) +
    geom_hline(yintercept = od_thresh, linetype = 2, color = "grey40") +
    geom_vline(xintercept = sd_thresh, linetype = 2, color = "grey40") +
    scale_color_manual(values = c("Injected outlier" = "#d73027",
                                  "Other machines" = "#1b9e77")) +
    scale_shape_manual(values = c("MCD set" = 16, "Flagged" = 17)) +
    labs(
      title = sprintf("Outlier map (q = %d PCs)", q),
      x = "Score distance",
      y = "Orthogonal distance",
      color = "",
      shape = "MCD weight"
    ) +
    theme_classic(base_size = 14)
  print(p_map)
  dev.off()

  extremes <- scores_df[order(-scores_df$dist), ]
  ext_names <- extremes$name[1:6]
  vars_pc1 <- names(sort(abs(loadings[, 1]), decreasing = TRUE))[1:5]
  vars_pc2 <- names(sort(abs(loadings[, 2]), decreasing = TRUE))[1:5]
  vars_key <- unique(c(vars_pc1, vars_pc2))
  X_ext <- X[ext_names, vars_key, drop = FALSE]

  cat("\nScores of the machines farthest in the PC1–PC2 space:\n")
  print(head(extremes[, c("PC1", "PC2", "group")], 6))

  cat("\nMachines farthest in the PC1–PC2 space:\n")
  print(ext_names)

  cat("\nVariables with the highest loadings on PC1:\n")
  print(vars_pc1)
```

robust_pca.R

```
  cat("\nVariables with the highest loadings on PC2:\n")
  print(vars_pc2)

  cat("\nOriginal values of these variables for each extreme machine:\n")
  print(round(X_ext, 3))

  cat("\nSaved robust PCA plots in folder:", out_dir, "\n")
  cat("Files:\n")
  cat("  01_scree_pve.png\n")
  cat("  02_cumulative_cpve.png\n")
  cat("  03_scores_pc1_pc2.png\n")
  cat("  04_loadings_pc1_pc2.png\n")
  cat("  05_biplot_pc1_pc2.png\n")
  cat("  06_outlier_map.png\n")
}

perform_robust_pca()
```

std_pca.R

```r
source("scripts/pca_original_scale.R")

X = load_machines_subset()

compute_pca(X, "plots/plots_pca_std", TRUE)
```