

NMDS CMJ Example

Nico Philipp PhD

2025-12-25

Non-Metric Multidimensional Scaling in Return to Play

An Example Using Countermovement Jump Data

The goal of this Non-Metric Multidimensional Scaling (nMDS) Analysis in a Return-to-Play (RTP) Context is to Visualize and Quantify How an Athlete's Overall Biomechanical Profile, Derived from Multiple Force-Plate Metrics Evolves During Rehab Relative to Their Healthy Baseline State.

Simulate Data

- Simulate CMJ Force-Time Metric Data Over a 6-month RTP Period
- Use Different Normalization Speeds to Simulate Metrics Becoming More “Normal”

```
# ---- 2) Simulate RTP period (6 months) with different normalization speeds ----
# RTP weekly testing over ~26 weeks
rtp_dates <- seq(from = as.Date("2025-01-01"), by = "1 week", length.out = 26)
t_weeks   <- 0:(length(rtp_dates) - 1)

# Define starting deviations (delta0) and half-lives by metric
# EARLY normalizers: shorter half-lives (2.5-3 wks)
# LATE normalizers: longer half-lives (12-16 wks)

cfg <- tribble(
  ~metric, ~delta0, ~half_life_weeks, ~noise_sd,
  # Early (fast convergence)
  "Jump Height (in)", -3.0, 3, 0.7,
  "Avg. Propulsive Force (N)", -350, 3, 105,

  # Late (slow convergence)
  "Avg. Braking Velocity (m/s)", 0.25, 12, 0.06,
  "Countermovement Depth (m)", 0.06, 12, 0.02,
  "Braking RFD (N/s)", -7000, 12, 1200,
  "Braking Force Asymmetry", 8.0, 16, 1.5,

  # Moderate
  "mRSI", -0.12, 6, 0.05,
  "Avg. Braking Force (N)", -250, 6, 100,
  "Propulsive Force Asymmetry", 3.5, 8, 1.2
)

# Build RTP table by applying decay_to_base metric-by-metric
rtp_mat <- baseline %>%
  left_join(cfg, by = "metric") %>%
  mutate(
    # impose a soft floor so Braking Force Asymmetry ends a bit > baseline
    # via long half-lif
    lift_tail = if_else(metric == "Braking Force Asymmetry", 1.2, 0), # ~+1.2% at end
    traj = pmap(
      list(mean, delta0, half_life_weeks, sd, lift_tail),
      function(m0, d0, hl, s, lt) {
        decay_to_base(t_weeks, base = m0, delta0 = d0, hl = hl, sd = s) +
          lt * (t_weeks / max(t_weeks)) # small linear tail-lift if applicable
      }
    )
  )

rtp_all_clean <- tibble(
  about = athlete_id,
```

```
    start_date = rtp_dates
  )

  for (i in seq_len(nrow(baseline))) {
    nm <- baseline$metric[i]
    rtp_all_clean[[nm]] <- rtp_mat$traj[[i]]
  }

  # Align names with objects
  cmj_all_clean <- rtp_all_clean
  comp_all_clean <- comp_all_clean
```

Prepare/Scale RTP and Comparison Datasets

- Convert All Metrics to Numeric
- Scaled each variable (mean = 0, SD = 1)
- Computed a Euclidean distance matrix between every pair of sessions — this quantifies how different one session's jump profile is from another's, across all metrics

```
# ---- 3) Analysis pipeline ----

# 1) Numeric metrics of interest (already defined as 'vars')
# 2) Prep both data sets, tag kind
rtp <- cmj_all_clean %>%
  mutate(start_date = parse_any_date(start_date),
         kind = "RTP") %>%
  select(about, start_date, kind, all_of(vars))

comp <- comp_all_clean %>%
  mutate(start_date = parse_any_date(start_date),
         kind = "Comparison") %>% # use "Game" later if desired
  select(about, start_date, kind, all_of(vars))

# 3) Combine, drop NA in vars, keep stable row id
df_an <- bind_rows(rtp, comp) %>%
  tidyr::drop_na(all_of(vars)) %>%
  mutate(.row_id = row_number()) %>%
  ungroup() %>%
  mutate(across(all_of(vars), ~ as.numeric(.))) # ensure numeric

# 4) Scale metrics + distance + nMDS
X_scaled <- df_an %>%
  select(all_of(vars)) %>%
  scale() %>%
  as.matrix()

set.seed(123)
D <- dist(X_scaled, method = "euclidean")
nmDS <- metaMDS(D, k = 2, trymax = 200, autotransform = FALSE)

## Run 0 stress 0.1117107
## Run 1 stress 0.1051861
## ... New best solution
## ... Procrustes: rmse 0.05322836 max resid 0.2704948
## Run 2 stress 0.1229023
## Run 3 stress 0.1082354
## Run 4 stress 0.105311
## ... Procrustes: rmse 0.007630033 max resid 0.03072063
## Run 5 stress 0.120002
## Run 6 stress 0.1265048
## Run 7 stress 0.1283335
## Run 8 stress 0.105311
## ... Procrustes: rmse 0.007637894 max resid 0.03073417
```

```
## Run 9 stress 0.1271301
## Run 10 stress 0.1155203
## Run 11 stress 0.1286858
## Run 12 stress 0.1082354
## Run 13 stress 0.1143945
## Run 14 stress 0.1117107
## Run 15 stress 0.1338474
## Run 16 stress 0.1317245
## Run 17 stress 0.1084034
## Run 18 stress 0.1162387
## Run 19 stress 0.105813
## Run 20 stress 0.1051764
## ... New best solution
## ... Procrustes: rmse 0.002268855 max resid 0.009515754
## ... Similar to previous best
## *** Best solution repeated 1 times
```

```
# 5) nMDS scores aligned to df_an
scores_df <- as.data.frame(scores(nmds, display = "sites")) %>%
  bind_cols(df_an %>% select(about, start_date, kind, .row_id)) %>%
  mutate(start_date = as.Date(start_date))

# 6) Convex hull for comparison sessions (only if >= 3 points)
comp_pts <- scores_df %>% filter(kind == "Comparison")
have_hull <- nrow(comp_pts) >= 3
if (have_hull) {
  hull_df <- comp_pts[ chull(comp_pts$NMDS1, comp_pts$NMDS2), ]
}
```

Perform Analysis and Plot NMDS Ordination vs. Healthy Baseline Sessions

- Close points = similar jump profiles
- Far points = different jump profiles
- The stress value measures how well the 2D configuration represents the full multivariate distances — values below 0.15 generally indicate a good fit
- Points Moving Closer To Baseline Over Time Suggests Vertical Stretch-Shortening Cycle Efficiency Increases Over The Course of Rehab, Which is the Goal

```
# 7) Plot (points only, centered title, real date legend)
# 7) Plot (points only, centered title, real date legend)
p <- ggplot(scores_df, aes(NMDS1, NMDS2)) +
  { if (have_hull)
    geom_polygon(data = hull_df, aes(NMDS1, NMDS2),
                inherit.aes = FALSE,
                fill = "darkgreen", alpha = 0.18, color = "darkgreen", linetype = "dashed") } +
  geom_point(data = comp_pts, aes(NMDS1, NMDS2),
            shape = 21, size = 4, fill = "darkgreen", color = "black") +
  geom_point(data = filter(scores_df, kind == "RTP"),
            aes(color = start_date), size = 3) +
  ggrepel::geom_label_repel(data = comp_pts,
                           aes(label = format(start_date, "%b %d")),
                           size = 3, fill = "white", label.size = 0.2,
                           min.segment.length = 0.1, seed = 1) +

  coord_equal() +
  scale_color_date(
    low = "red", high = "green",
    name = "Date", labels = date_format("%b %d"),
    breaks = pretty_breaks(n = 6)
  ) +
  guides(color = guide_colorbar(barwidth = 18, barheight = 0.6)) +
  theme_classic(base_size = 13) +
  theme(legend.position = "bottom",
        plot.title = element_text(hjust = 0.5, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5)) +
  labs(title = "RTP sessions vs comparison 'healthy baseline' profile (nMDS)",
        subtitle = sprintf("Stress = %.3f (Euclidean on scaled metrics)", nmds$stress),
        x = "NMDS1", y = "NMDS2")

# Get NMDS coordinate ranges
x_range <- range(scores_df$NMDS1)
y_range <- range(scores_df$NMDS2)

# Compute adaptive offsets based on axis range
x_offset <- 0.1 * diff(x_range) # horizontal shift (~10% of width)
y_offset <- 0.15 * diff(y_range) # vertical shift (~15% of height)

# Far-left label (First Rehab Assessment)
label_left_x <- x_range[1] - 0.1 * diff(x_range) + x_offset
label_left_y <- mean(y_range) + y_offset
```

```

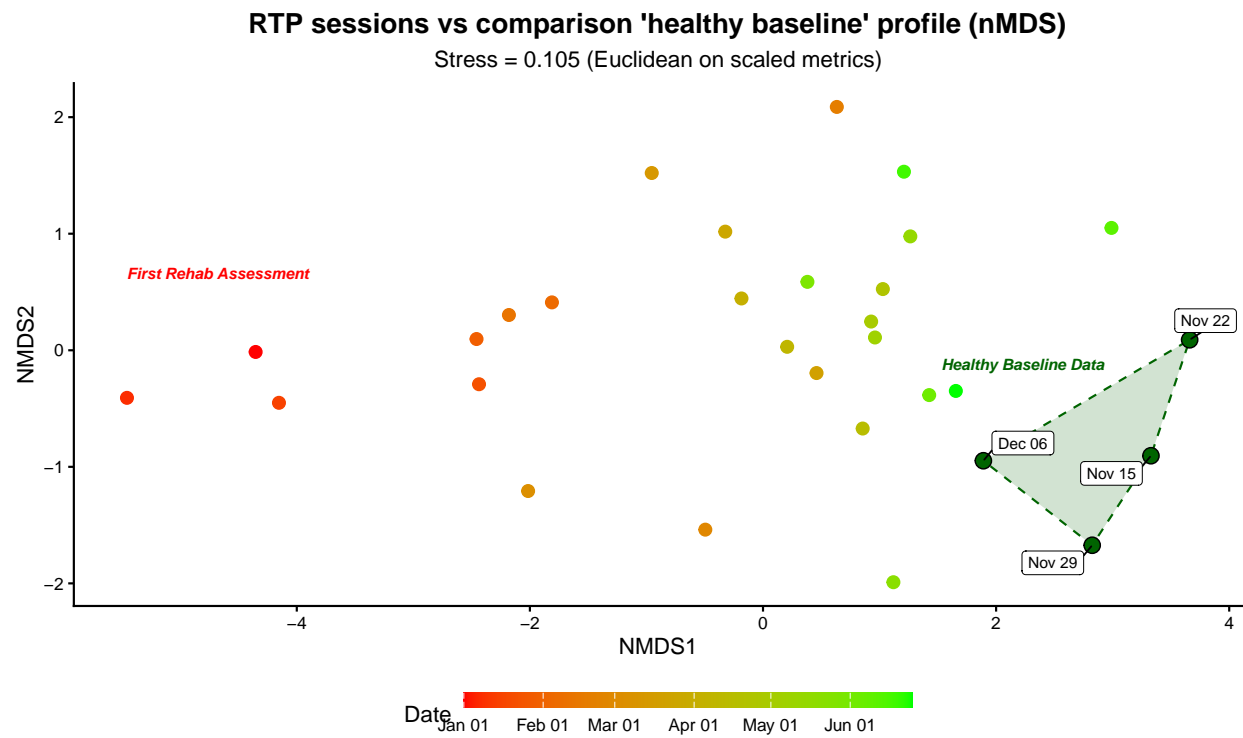
# Hull centroid (Healthy Baseline Data)
hull_center <- comp_pts %>%
  summarise(cx = mean(NMDS1), cy = mean(NMDS2)) %>%
  as.list()

label_hull_x <- hull_center$cx + x_offset / 2
label_hull_y <- hull_center$cy + y_offset

# Add to your ggplot
p <- p +
  annotate("text",
    x = label_left_x, y = label_left_y,
    label = "First Rehab Assessment",
    size = 3, fontface = "bold.italic",
    color = "red", hjust = 0) +
  annotate("text",
    x = hull_center$cx, y = hull_center$cy + 0.5, # small vertical offset
    label = "Healthy Baseline Data",
    size = 3, fontface = "bold.italic",
    color = "darkgreen", hjust = 1, vjust = -2)

print(p)

```



Quantify Similarity

- Compute Individual Distances For RTP Sessions Compared to Healthy Baseline
- The Convergence Plot Quantifies the Same Story Shown in the NMDS Ordination — The Athlete is Getting Closer to Baseline Mechanically as Rehab Progresses.
- Stretch-Shortening Cycle Efficiency Progresses Early Then Plateaus Toward the End

```
# 8) Quantify similarity: distances from each RTP session to all Comparison sessions
dist_mat <- as.matrix(D)
rtp_idx <- which(df_an$kind == "RTP")
comp_idx <- which(df_an$kind == "Comparison")

rtp_dist_tbl <- tibble(
  about      = df_an$about[rtp_idx],
  date       = df_an$start_date[rtp_idx],
  mean_dist_to_comp = rowMeans(dist_mat[rtp_idx, comp_idx, drop = FALSE], na.rm = TRUE),
  median_dist_to_comp = apply(dist_mat[rtp_idx, comp_idx, drop = FALSE], 1, median, na.rm = TRUE),
  min_dist_to_comp   = apply(dist_mat[rtp_idx, comp_idx, drop = FALSE], 1, min, na.rm = TRUE)
) %>%
  arrange(date)

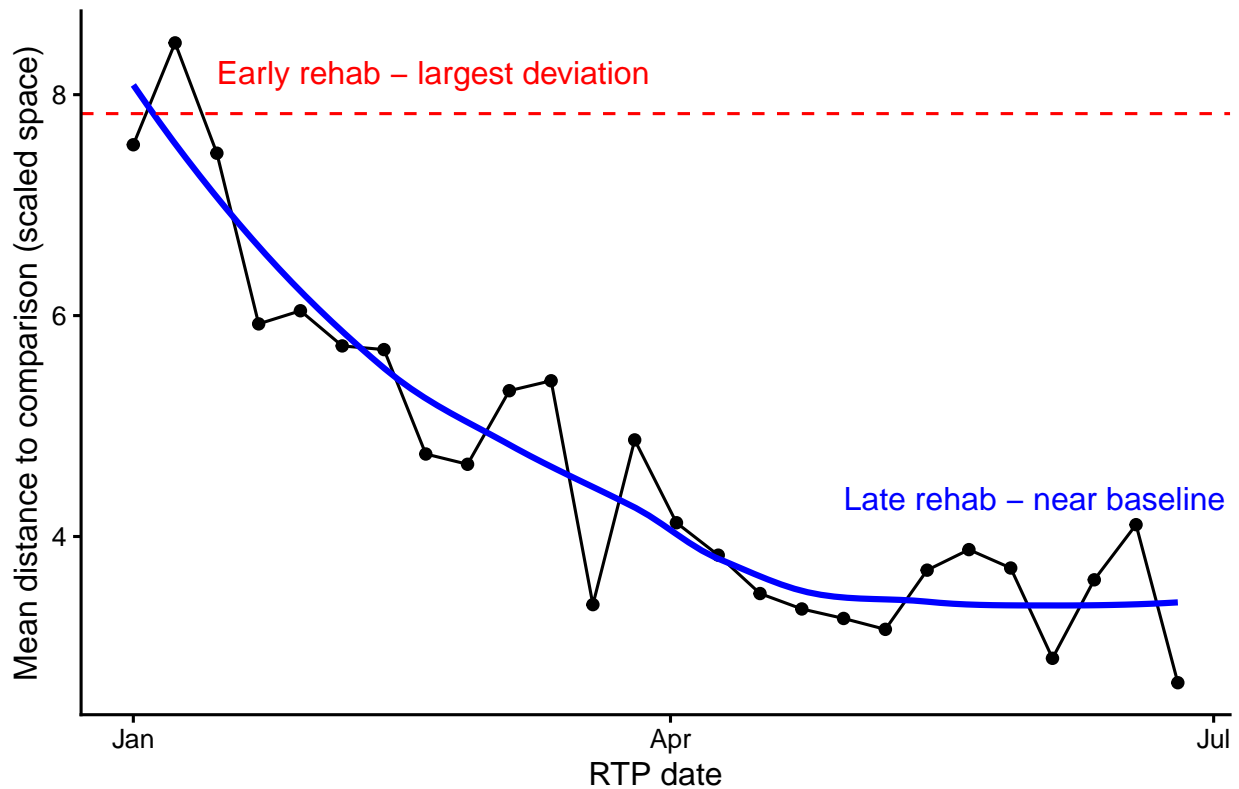
# Optional overall summary
overall_dist_summary <- rtp_dist_tbl %>%
  summarise(
    mean_of_means = mean(mean_dist_to_comp),
    median_of_means = median(median_dist_to_comp),
    min_of_means = min(min_dist_to_comp),
    max_of_means = max(max_dist_to_comp)
  )

#rtp_dist_tbl
#overall_dist_summary

# Visual check of convergence over the 6 months
ggplot(rtp_dist_tbl, aes(date, mean_dist_to_comp)) +
  geom_line() + geom_point() +
  theme_classic(base_size = 12) +
  labs(title = "Convergence of RTP profiles toward comparison baseline",
       y = "Mean distance to comparison (scaled space)", x = "RTP date") +
  geom_smooth(method = "loess", se = FALSE, color = "blue") +
  geom_hline(yintercept = mean(rtp_dist_tbl$mean_dist_to_comp[1:3]),
            linetype = "dashed", color = "red") +
  annotate("text", x = as.Date("2025-01-15"), y = 8.2,
          label = "Early rehab - largest deviation", hjust = 0, color = "red") +
  annotate("text", x = as.Date("2025-06-01"), y = 3.2,
          label = "Late rehab - near baseline", hjust = 0.5, vjust = -5, color = "blue")

## 'geom_smooth()' using formula = 'y ~ x'
```

Convergence of RTP profiles toward comparison baseline



```
# 9) Which metrics drive separation? envfit vectors
fit <- envfit(nmds, df_an %>% select(all_of(vars)), permutations = 999)
vec_tbl <- as.data.frame(scores(fit, "vectors"))
vec_tbl$Variable <- rownames(vec_tbl)
vec_tbl$R2 <- fit$vectors$r
vec_tbl$P <- fit$vectors$pvals
vec_tbl <- vec_tbl %>% select(Variable, NMDS1, NMDS2, R2, P) %>% arrange(P)
#vec_tbl

# ---- Overlay envfit vectors on the nMDS plot ----
xr <- range(scores_df$NMDS1); yr <- range(scores_df$NMDS2)
scale_fac <- 0.8 * min(diff(xr), diff(yr))

vec <- as.data.frame(scores(fit, "vectors"))
vec$Variable <- rownames(vec)
vec$R2 <- fit$vectors$r
vec$P <- fit$vectors$pvals

# keep only significant variables for clarity
vec <- vec %>% filter(P <= 0.05)

vec$wNMDS1 <- vec$NMDS1 * scale_fac * sqrt(vec$R2)
vec$wNMDS2 <- vec$NMDS2 * scale_fac * sqrt(vec$R2)
```

Plot NMDS Ordination + Vectors Representing Metrics

- Which Metrics Drive Separation?
- Arrows Pointing Toward the Baseline Represent Metrics That Were Lower Early in RTP and Increased Toward Baseline (e.g., Braking RFD)
- Arrows Pointing Away from Baseline Represent Metrics That Were Elevated Early in RTP (e.g., Braking Force Asymmetry)
- Arrow Length = R-Squared Highlighting How well does each variable explain the spatial arrangement of sessions in this nMDS space?

```
p_envfit <- p +  
  geom_segment(  
    data = vec,  
    aes(x = 0, y = 0, xend = wNMDS1, yend = wNMDS2),  
    inherit.aes = FALSE,  
    color = "grey25",  
    linewidth = 0.7,  
    arrow = grid::arrow(length = grid::unit(0.18, "cm"))  
  ) +  
  geom_text_repel(  
    data = vec,  
    aes(x = wNMDS1, y = wNMDS2, label = paste0(Variable, "\nR2=", round(R2, 2))),  
    inherit.aes = FALSE,  
    size = 3,  
    color = "grey10",  
    box.padding = 0.35,  
    point.padding = 0.2,  
    min.segment.length = 0,  
    max.overlaps = Inf  
  )  
  
print(p_envfit)
```

