Priority Fixes and Improvements for `subscreenfunnel` Code

July 10, 2025

Based on the paper and the code analysis, here are the **priority fixes** and **realistic improvements** for the `subscreenfunnel` code:

1 Priority 1: Critical Statistical Fixes (Must Fix)

1.1 1. Fix the Fundamental Permutation Logic

```r
# CURRENT (WRONG): Resampling subjects
all_samples <- replicate(nperm, slice_sample(data_trimmed, n = sampsize
    ))

# CORRECT: Permute treatment assignments within subgroups
generate_permuted_data <- function(data, treat, nperm) {
  lapply(1:nperm, function(i) {
    permuted_data <- data
    permuted_data[,treat] <- sample(data[,treat])  # Permute treatment
        labels
    return(permuted_data)
  })
}
```

Why this is critical: The current implementation invalidates the entire statistical foundation of the method described in the paper.

1.2 2. Add Missing LOESS Smoothing

```r
# Add after quantile calculation
smooth_funnel_bounds <- function(support_points, lower_bound, upper_
    bound) {
  # LOESS smoothing as described in paper (span = 0.25)
  loess_lower <- loess(lower_bound ~ support_points, span = 0.25)
  loess_upper <- loess(upper_bound ~ support_points, span = 0.25)

  list(
    support_points = support_points,
    lower_smooth = predict(loess_lower),
    upper_smooth = predict(loess_upper)
  )
}
```

Why this is critical: The paper explicitly states this is essential for creating the funnel boundaries.

2 Priority 2: Parameter Validation and Error Handling

2.1 3. Robust Parameter Validation

```r
validate_funnel_parameters <- function(data, treat, endpoints, nperm, alpha) {
  # Check data columns
  if (!treat %in% colnames(data)) {
    stop("Treatment column '", treat, "' not found. Available: ",
         paste(colnames(data), collapse = ", "))
  }

  # Validate sample sizes
  treatment_counts <- table(data[,treat])
  if (any(treatment_counts < 10)) {
    warning("Small treatment groups detected. Results may be unstable.")
  }

  # Validate permutation parameters
  if (nperm < 1000) {
    warning("nperm < 1000 may produce unstable confidence intervals. Paper recommends   1000  .")
  }

  if (alpha < 0.001 || alpha > 0.2) {
    warning("Unusual alpha value. Typical range: 0.001 to 0.1")
  }
}
```

2.2 4. Fix Support Point Generation

```r
# Current implementation has off-by-one error
generate_support_points <- function(min_size, max_size, n_points = 50) {
  # Paper specifies exactly 50 points, not 51
  sqrt_min <- sqrt(max(min_size, 4))  # Minimum of 4 subjects
  sqrt_max <- sqrt(max_size)

  sqrt_seq <- seq(sqrt_min, sqrt_max, length.out = n_points)
  support_points <- round(sqrt_seq^2)

  # Remove duplicates and ensure minimum size
  unique(pmax(support_points, 4))
}
```

3 Priority 3: Performance and Memory Optimization

3.1 5. Memory-Efficient Permutation

```r
# Instead of storing all permuted datasets
efficient_permutation <- function(data, treat, eval_function, support_
    points, nperm) {

  results <- vector("list", length(support_points))

  for (i in seq_along(support_points)) {
    n_size <- support_points[i]
    perm_results <- numeric(nperm)

    for (perm in 1:nperm) {
      # Sample subset first, then permute
      subset_indices <- sample(nrow(data), min(n_size, nrow(data)))
      subset_data <- data[subset_indices, ]

      # Permute treatment in subset
      subset_data[,treat] <- sample(subset_data[,treat])

      # Calculate result immediately (don't store data)
      perm_results[perm] <- eval_function(subset_data)[[1]]  # Assuming
          single endpoint
    }

    results[[i]] <- quantile(perm_results, probs = c(alpha/2, 1-alpha/
        2), na.rm = TRUE)
  }

  return(results)
}
```

4 Priority 4: Realistic Enhancements

4.1 6. Better Default Parameters

```r
subscreenfunnel_improved <- function(
  data,
  H,
  eval_function,
  min_start = 10,         # Increased from 2 (more stable)
  n_support_points = 50,  # Fixed to match paper exactly
  nperm = 1000,           # Increased from 200 (paper recommendation)
  alpha = 0.05,           # More practical than 0.001
  stratified = TRUE,
  treat = "treatment",    # More standard name
  endpoints = NULL,       # Auto-detect from eval_function
  verbose = TRUE,
  nkernel = min(4, parallel::detectCores() - 1)  # Better default
) {
  # Implementation...
}
```

4.2 7. Add Convergence Diagnostics

```
check_convergence <- function(perm_results, nperm) {
  # Check if we have enough permutations for stable quantiles
  if (nperm < 1000) {
    warning("Consider increasing nperm for more stable results")
  }

  # Check for too many NAs
  na_prop <- mean(is.na(perm_results))
  if (na_prop > 0.1) {
    warning("High proportion of NA results (", round(na_prop*100, 1), "
      %). Check eval_function.")
  }

  # Simple stability check (compare first and second half)
  if (nperm >= 500) {
    half1 <- quantile(perm_results[1:(nperm/2)], c(0.025, 0.975), na.rm
       = TRUE)
    half2 <- quantile(perm_results[(nperm/2+1):nperm], c(0.025, 0.975),
       na.rm = TRUE)

    if (max(abs(half1 - half2)) > 0.1) {
      warning("Quantiles appear unstable. Consider increasing nperm.")
    }
  }
}
```

4.3 8. Improved Output Structure

```
# Return more informative results
create_funnel_output <- function(H, support_points, quantiles_smooth,
    alpha, nperm) {
  funnel_data <- data.frame(
    n = support_points,
    lower_bound = quantiles_smooth$lower_smooth,
    upper_bound = quantiles_smooth$upper_smooth,
    alpha = alpha
  )

  H$funnel <- list(
    bounds = funnel_data,
    parameters = list(
      alpha = alpha,
      nperm = nperm,
      method = "permutation_loess",
      span = 0.25
```

```r
    ),
    diagnostics = list(
      n_support_points = length(support_points),
      min_subgroup_size = min(support_points),
      max_subgroup_size = max(support_points)
    )
  )

  class(H$funnel) <- "SubScreenFunnel"
  return(H)
}
```

5 Implementation Priority Order

1. **Week 1**: Fix permutation logic (#1) - This is critical for validity
2. **Week 2**: Add LOESS smoothing (#2) - Essential for matching paper
3. **Week 3**: Parameter validation (#3) - Prevents user errors
4. **Week 4**: Fix support points (#4) - Ensures correct implementation
5. **Week 5**: Memory optimization (#5) - Improves scalability
6. **Week 6**: Better defaults and diagnostics (#6-8) - Improves usability

6 Quick Win Implementation

For immediate improvement, focus on this minimal fix:

```r
# Minimal critical fix
subscreenfunnel_quickfix <- function(data, H, eval_function, treat = "
    treatment",
                                     endpoints = NULL, nperm = 1000,
                                        alpha = 0.05, ...) {

  # 1. Validate inputs
  if (!treat %in% colnames(data)) {
    stop("Treatment column not found: ", treat)
  }

  # 2. Fix permutation (most critical)
  permuted_results <- replicate(nperm, {
    perm_data <- data
    perm_data[,treat] <- sample(data[,treat])  # Correct permutation
    eval_function(perm_data)[[1]]  # Get first endpoint
  })

  # 3. Calculate quantiles
  bounds <- quantile(permuted_results, c(alpha/2, 1-alpha/2), na.rm =
     TRUE)

  # 4. Add to results
  H$funnel_bounds <- bounds
  return(H)
}
```

This addresses the most critical statistical flaw while being implementable immediately.