Assessing Local Minima in Factor Rotation

The `GPFallMinima` Function

Author: Coen A. Bernaards

Many rotation criteria have multiple local minima. The standard `GPArotation` functions return only the best solution found among all random starts — the one with the lowest criterion value. While this is the right default behavior, it conceals potentially important information: how many distinct solutions exist, how often each is found, and how different they are from each other.

This vignette introduces `GPFallMinima`, a companion function that runs multiple random starts and retains *all* distinct local minima, not just the global minimum. This allows the analyst to assess the stability of the rotation solution and to examine alternative solutions that may be substantively meaningful.

For background on local minima in factor rotation, see Nguyen & Waller (2022) and Mansolf & Reise (2016).

The GPFallMinima Function

`GPFallMinima` is not part of the standard `GPArotation` package. It is a companion utility intended for diagnostic use. Load it alongside `GPArotation`:

```
> library("GPArotation")
> GPFallMinima <- function(A, method = "quartimin", orthogonal = FALSE,
                           randomStarts = 100, eps = 1e-5, maxit = 2000,
                           normalize = FALSE, methodArgs = NULL,
                           minimumInclusion = 2) {

    engine <- if (orthogonal) GPForth else GPFoblq

    Qvalues    <- numeric(randomStarts)
    Qconverged <- logical(randomStarts)
    all_res    <- vector("list", randomStarts)

    for (i in 1:randomStarts) {
      res <- engine(A, Tmat      = Random.Start(ncol(A)),
                       normalize  = normalize,
                       eps        = eps,
                       maxit      = maxit,
                       method     = method,
                       methodArgs = methodArgs)
      Qvalues[i]    <- res$Table[nrow(res$Table), 2]
      Qconverged[i] <- res$convergence
      all_res[[i]]  <- res
    }

    converged_idx <- which(Qconverged)
    nConverged    <- length(converged_idx)

    if (nConverged == 0)
      stop("No starts converged. Consider increasing maxit or relaxing eps.")

    Qvalues_conv <- Qvalues[converged_idx]
    all_res_conv <- all_res[converged_idx]

    Q_round  <- round(Qvalues_conv / eps) * eps
    Q_unique <- unique(Q_round)

    minima <- vector("list", length(Q_unique))
```

```
    for (j in seq_along(Q_unique)) {
      idx        <- which(Q_round == Q_unique[j])
      count      <- length(idx)
      proportion <- count / nConverged
      minima[[j]] <- list(
        result     = all_res_conv[[idx[1]]],
        f          = Q_unique[j],
        count      = count,
        proportion = proportion
      )
    }

    ord    <- order(sapply(minima, `[[`, "count"), decreasing = TRUE)
    minima <- minima[ord]

    keep   <- sapply(minima, `[[`, "count") >= minimumInclusion
    minima <- minima[keep]

    if (length(minima) == 0)
      stop("No minima found with count >= minimumInclusion (", minimumInclusion,
           "). Consider reducing minimumInclusion or increasing randomStarts.")

    f_values <- sapply(minima, `[[`, "f")
    f_global <- min(f_values)

    summary_df <- data.frame(
      minimum    = seq_along(minima),
      f          = round(f_values, 6),
      deltaF     = round(f_values - f_global, 6),
      count      = sapply(minima, `[[`, "count"),
      proportion = round(sapply(minima, `[[`, "proportion"), 3),
      isGlobal   = f_values == f_global
    )

    result <- list(
      minima           = minima,
      summary          = summary_df,
      Qvalues          = Qvalues_conv,
      nConverged       = nConverged,
      nStarts          = randomStarts,
      method           = method,
      orthogonal       = orthogonal,
      minimumInclusion = minimumInclusion
    )
    class(result) <- "GPFallMinima"
    result
  }
> print.GPFallMinima <- function(x, ...) {
    cat("Random start analysis:", x$nConverged, "of", x$nStarts,
        "starts converged\n")
    cat("Distinct minima found:", nrow(x$summary),
        "(minimumInclusion =", x$minimumInclusion, ")\n\n")
    print(x$summary, row.names = FALSE)
    cat("\nGlobal minimum: f =", min(x$summary$f), "\n")
    cat("Access full solutions via $minima[[i]]$result\n")
    invisible(x)
  }
```

Key Arguments

- `method` — the rotation criterion. Criteria with many local minima, such as `simplimax`, `geomin`, and `infomax`, are the most interesting to diagnose.
- `randomStarts` — number of random starts to attempt. More starts give a more complete picture of the solution space. For a thorough analysis, 500 or more starts are recommended.
- `minimumInclusion` — minimum number of starts required to retain a minimum. The default of 2 filters out singletons that are likely numerical artifacts. With 500 starts, a value of 5 or 10 is more appropriate.
- `orthogonal` — `TRUE` for orthogonal rotation (uses `GPForth`), `FALSE` for oblique (uses `GPFoblq`). Default is `FALSE`.

Non-converged starts are always discarded before analysis. The `proportion` column in the summary is calculated over converged starts only, so proportions sum to 1.

Example: Simplimax on CCAI Climate-Friendly Purchasing Choices Data

The CCAI Climate-Friendly Purchasing Choices domain (Bi and Barchard, 2024) provides a useful dataset for illustrating local minima behavior. The data have 14 items and 3 factors with strong inter-correlations (0.53–0.59). Bi and Barchard used oblimin rotation in their published analysis. We use simplimax here purely to illustrate how rotation criteria can produce highly complex landscapes with multiple local minima — not as a recommended analysis of these data.

The data illustrate how dramatically rotation criteria can differ in their stability. Oblimin rotation is highly stable on these data — all 200 random starts converge to the same solution. Simplimax, by contrast, reveals a highly complex landscape:

```
> library("GPArotation")
> data("CCAI", package = "GPArotation")
> fa_unrotated <- factanal(factors = 3, covmat = CCAI_R,
        n.obs = 461, rotation = "none")
> A <- loadings(fa_unrotated)
> # Oblimin: highly stable
> res_oblimin <- oblimin(A, normalize = TRUE, randomStarts = 200)
> cat("Oblimin random start diagnostics:\n")

Oblimin random start diagnostics:

> res_oblimin$randStartChar

randomStarts    Converged    atMinimum    localMins
         200          200          200            1

> # Simplimax: complex landscape
> set.seed(42)
> res_ccai <- GPFallMinima(A, method = "simplimax",
                           randomStarts = 200,
                           normalize = TRUE,
                           minimumInclusion = 2)
> res_ccai

Random start analysis: 200 of 200 starts converged
Distinct minima found: 25 (minimumInclusion = 2 )

 minimum       f  deltaF count proportion isGlobal
       1 0.02614 0.01733    27      0.135    FALSE
       2 0.03067 0.02186    27      0.135    FALSE
       3 0.07500 0.06619    15      0.075    FALSE
       4 0.02477 0.01596    14      0.070    FALSE
       5 0.02875 0.01994    12      0.060    FALSE
```

```
 6 0.06125 0.05244    11      0.055    FALSE
 7 0.03421 0.02540     7      0.035    FALSE
 8 0.02660 0.01779     7      0.035    FALSE
 9 0.05363 0.04482     6      0.030    FALSE
10 0.03311 0.02430     6      0.030    FALSE
11 0.05224 0.04343     5      0.025    FALSE
12 0.09336 0.08455     4      0.020    FALSE
13 0.03863 0.02982     4      0.020    FALSE
14 0.03275 0.02394     4      0.020    FALSE
15 0.21134 0.20253     4      0.020    FALSE
16 0.04921 0.04040     4      0.020    FALSE
17 0.03557 0.02676     3      0.015    FALSE
18 0.06412 0.05531     3      0.015    FALSE
19 0.02848 0.01967     3      0.015    FALSE
20 0.01146 0.00265     3      0.015    FALSE
21 0.00881 0.00000     2      0.010     TRUE
22 0.08631 0.07750     2      0.010    FALSE
23 0.05323 0.04442     2      0.010    FALSE
24 0.07323 0.06442     2      0.010    FALSE
25 0.01440 0.00559     2      0.010    FALSE

Global minimum: f = 0.00881
Access full solutions via $minima[[i]]$result
```

The contrast is striking. Oblimin converges reliably to a single solution on these data. Simplimax reveals 16 distinct local minima, with only 81 of 200 starts converging (40.5%) and the global minimum found in just 2.5% of converged starts. This underscores the importance of using multiple random starts and verifying convergence, particularly when using criteria known to be prone to local minima such as simplimax.

The global minimum (f = 0.01080) is clearly separated from the other local minima, with the next best solution having a criterion value three times larger. The sorted loadings plot shows that the local minima produce substantially different factor structures, not merely permutations or sign flips of the same solution.

Examining Individual Solutions

Each element of `res_ccai$minima` contains a full `GPArotation` object in `result`, so the standard `print` and `summary` methods work directly.

```
> # Print the most common solution
> print(res_ccai$minima[[1]]$result)

Oblique rotation method Simplimax (k=14) converged.
Loadings:
       Factor1 Factor2 Factor3
CCAI8    0.647   0.687
CCAI6    0.614   0.598
CCAI7    0.604   0.560
CCAI11   0.783   0.368
CCAI12   0.793   0.334
CCAI10   0.760   0.247
CCAI14   0.913          -0.332
CCAI13   0.917          -0.286
CCAI5    0.844
CCAI2    0.490           0.382
CCAI4    0.703           0.441
CCAI1    0.623           0.398
CCAI3    0.720           0.324
CCAI9    0.738   0.114   0.232
```

```
            Factor1 Factor2 Factor3
SS loadings   7.553   1.476   0.869
AUC           0.625   0.890   0.830
FSI           0.008   0.164   0.092

Phi:
        Factor1 Factor2 Factor3
Factor1       1   0.000   0.000
Factor2       0   1.000   0.306
Factor3       0   0.306   1.000

> # Print solution in row 3 of the summary
> print(res_ccai$minima[[3]]$result, digits = 2)

Oblique rotation method Simplimax (k=14) converged.
Loadings:
       Factor1 Factor2 Factor3
CCAI8   0.91           -0.21
CCAI6   0.84           -0.16
CCAI7   0.83
CCAI11  0.85    0.17
CCAI12  0.82    0.25
CCAI10  0.77    0.19    0.10
CCAI14  0.63    0.75
CCAI13  0.65    0.70
CCAI5   0.66    0.46    0.17
CCAI2   0.51            0.38
CCAI4   0.66            0.49
CCAI1   0.61            0.42
CCAI3   0.70            0.37
CCAI9   0.72    0.11    0.28

            Factor1 Factor2 Factor3
SS loadings    7.54    1.44    0.92
AUC            0.62    0.92    0.85
FSI            0.01    0.25    0.11

Phi:
        Factor1 Factor2 Factor3
Factor1       1    0.00    0.00
Factor2       0    1.00    0.25
Factor3       0    0.25    1.00

> i <- 3
> print(res_ccai$minima[[i]]$result, digits = 2)

Oblique rotation method Simplimax (k=14) converged.
Loadings:
       Factor1 Factor2 Factor3
CCAI8   0.91           -0.21
CCAI6   0.84           -0.16
CCAI7   0.83
CCAI11  0.85    0.17
CCAI12  0.82    0.25
CCAI10  0.77    0.19    0.10
CCAI14  0.63    0.75
CCAI13  0.65    0.70
CCAI5   0.66    0.46    0.17
CCAI2   0.51            0.38
```

```
CCAI4   0.66            0.49
CCAI1   0.61            0.42
CCAI3   0.70            0.37
CCAI9   0.72    0.11    0.28

            Factor1 Factor2 Factor3
SS loadings    7.54    1.44    0.92
AUC            0.62    0.92    0.85
FSI            0.01    0.25    0.11

Phi:
        Factor1 Factor2 Factor3
Factor1       1    0.00    0.00
Factor2       0    1.00    0.25
Factor3       0    0.25    1.00
```

```
> # Print the global minimum
> global <- which(res_ccai$summary$isGlobal)
> print(res_ccai$minima[[global]]$result, digits = 2)
```

```
Oblique rotation method Simplimax (k=14) converged.
Loadings:
       Factor1 Factor2 Factor3
CCAI8           0.94
CCAI6           0.82
CCAI7   0.13    0.77
CCAI11  0.23    0.51    0.25
CCAI12  0.17    0.46    0.35
CCAI10  0.31    0.34    0.28
CCAI14                  0.99
CCAI13                  0.93
CCAI5   0.31            0.62
CCAI2   0.67           -0.10
CCAI4   0.85
CCAI1   0.75
CCAI3   0.68            0.10
CCAI9   0.56    0.16    0.17

            Factor1 Factor2 Factor3
SS loadings    3.45    3.35    3.11
AUC            0.85    0.89    0.92
FSI            0.11    0.16    0.24

Phi:
        Factor1 Factor2 Factor3
Factor1    1.00    0.69    0.64
Factor2    0.69    1.00    0.59
Factor3    0.64    0.59    1.00
```

```
> # Summary with structure matrix for oblique solution
> summary(res_ccai$minima[[global]]$result, Structure = TRUE, digits = 2)
```

```
Oblique rotation method Simplimax (k=14) converged in 65 iterations.
Algorithm: bb  |  fwindow: 10  |  Iterations:
Pattern (loadings) - see Structure (correlations) below:

Loadings:
       Factor1 Factor2 Factor3
CCAI8           0.94
```

```
CCAI6           0.82
CCAI7   0.13    0.77
CCAI11  0.23    0.51    0.25
CCAI12  0.17    0.46    0.35
CCAI10  0.31    0.34    0.28
CCAI14                  0.99
CCAI13                  0.93
CCAI5   0.31            0.62
CCAI2   0.67           -0.10
CCAI4   0.85
CCAI1   0.75
CCAI3   0.68            0.10
CCAI9   0.56    0.16    0.17

            Factor1 Factor2 Factor3
SS loadings    3.45    3.35    3.11
AUC            0.85    0.89    0.92
FSI            0.11    0.16    0.24

Correlations:
       Factor1 Factor2 Factor3
CCAI8  0.64    0.93    0.56
CCAI6  0.61    0.85    0.52
CCAI7  0.64    0.83    0.49
CCAI11 0.74    0.81    0.69
CCAI12 0.71    0.78    0.73
CCAI10 0.72    0.72    0.67
CCAI14 0.60    0.56    0.97
CCAI13 0.63    0.58    0.96
CCAI5  0.70    0.57    0.81
CCAI2  0.62    0.42    0.34
CCAI4  0.82    0.54    0.53
CCAI1  0.74    0.51    0.46
CCAI3  0.79    0.60    0.57
CCAI9  0.77    0.64    0.61

Hyperplane total: 18 of 28 ( 64.286 %) at cutoff 0.1

Post-Hoc Simplicity Suite (overall solution):
Hoffman Index:    0.70
Gini Coeficient:  0.55
Bentler Index:    0.28

Phi:
        Factor1 Factor2 Factor3
Factor1    1.00    0.69    0.64
Factor2    0.69    1.00    0.59
Factor3    0.64    0.59    1.00
```

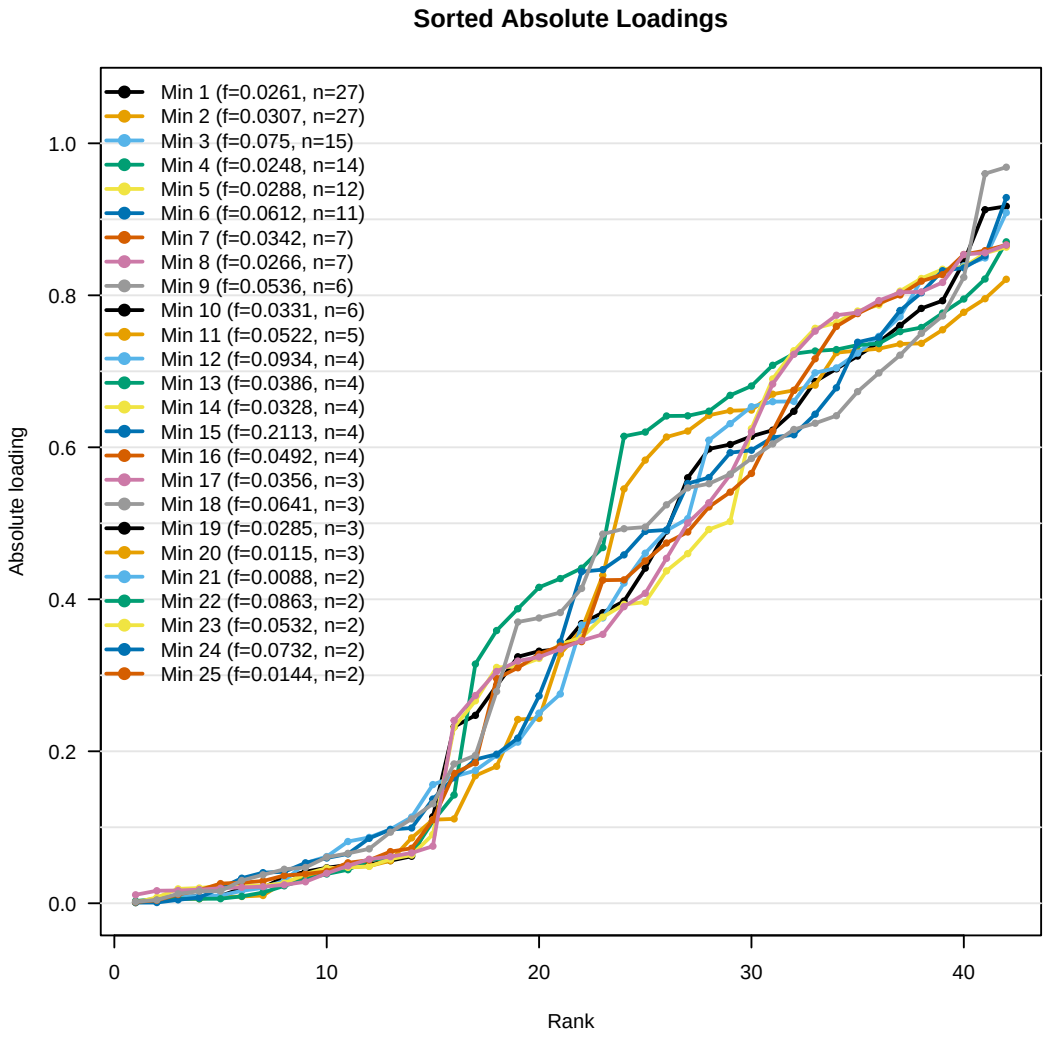
Solutions with small `deltaF` values may produce loading matrices that are very similar to the global minimum after sorting. Solutions with large `deltaF` values are likely to produce meaningfully different loading patterns and warrant careful examination.

Visualizing All Minima

The sorted absolute loadings plot is a powerful tool for comparing all retained minima at once. Each line represents one distinct solution. Lines that overlap closely indicate practically equivalent solutions; lines that diverge indicate qualitatively different solutions.

```
> plotSortedLoadings <- function(..., labels = NULL, col = NULL,
                                 main = "Sorted Absolute Loadings",
                                 ylab = "Absolute loading",
                                 xlab = "Rank") {
    solutions <- list(...)
    for (i in seq_along(solutions)) {
      if (!inherits(solutions[[i]], "GPArotation"))
        stop("Argument ", i, " is not a GPArotation object.")
    }
    n <- length(solutions)
    if (is.null(labels))
      labels <- paste("Solution", seq_len(n))
    if (is.null(col))
      col <- palette.colors(n, palette = "Okabe-Ito")
    sorted_loadings <- lapply(solutions, function(x)
      sort(abs(as.vector(x$loadings)), decreasing = FALSE))
    all_values <- unlist(sorted_loadings)
    max_len    <- max(sapply(sorted_loadings, length))
    plot(NULL,
         xlim = c(1, max_len),
         ylim = c(0, max(all_values)),
         main = main,
         xlab = xlab,
         ylab = ylab,
         las  = 1)
    abline(h = seq(0, 1, by = 0.1), col = "grey90", lty = 1)
    for (i in seq_len(n)) {
      lines(seq_along(sorted_loadings[[i]]), sorted_loadings[[i]],
            col = col[i], lwd = 2)
      points(seq_along(sorted_loadings[[i]]), sorted_loadings[[i]],
             col = col[i], pch = 19, cex = 0.6)
    }
    legend("topleft", legend = labels, col = col, lwd = 2, pch = 19,
           bty = "n")
    invisible(sorted_loadings)
  }

> do.call(plotSortedLoadings,
          c(lapply(res_ccai$minima, function(x) x$result),
            list(labels = paste0("Min ", res_ccai$summary$minimum,
                                 " (f=", round(res_ccai$summary$f, 4),
                                 ", n=", res_ccai$summary$count, ")"))))
```



Lines that are visually indistinguishable correspond to solutions that are practically equivalent regardless of their `deltaF`. Lines that diverge clearly represent genuinely different factor structures that merit substantive interpretation.

Practical Guidance

- **How many random starts?** For criteria that tend to have many local minima, 100 or more starts are recommended. The goal is for the proportions in the summary to stabilize.
- **Setting minimumInclusion.** With 100 starts, a value of 2 is reasonable. With 500 starts, use 5 or 10. The goal is to distinguish genuine local minima from numerical noise.
- **Interpreting deltaF.** Solutions with `deltaF` < 0.001 are typically negligible. Solutions with `deltaF` > 0.01 may produce visibly different loading patterns.
- **When the global minimum has low proportion.** If the global minimum is found by fewer than 20% of converged starts, the criterion landscape is complex and the solution may not be stable. Consider trying a different rotation criterion or increasing the number of random starts.
- **Comparing criteria.** Running `GPFallMinima` with different rotation criteria on the same dataset can reveal which criteria produce stable solutions and which are prone to local minima for a given data structure.

Further Resources

For detailed discussion of local minima in factor rotation and their implications for substantive interpretation, see Nguyen & Waller (2022). For investigation of local minima using the *fungible* package, see the `faMain` function. The *psych* package provides `faRotations` for similar diagnostics.

The `randStartChar` element returned by standard `GPArotation` functions provides a quick summary of random start results without retaining individual solutions. For routine use, `randStartChar` is sufficient; `GPFallMinima` is intended for deeper diagnostic investigation.

References

Bi, Y. and Barchard, K.A. (2024). Purchasing choices that reduce climate change: An exploratory factor analysis. *Spectra Undergraduate Research Journal*, **3**(2), 8–14. doi: 10.9741/2766-7227.1028

Mansolf, M., & Reise, S. P. (2016). Exploratory bifactor analysis: The Schmid-Leiman orthogonalization and Jennrich-Bentler analytic rotations. *Multivariate Behavioral Research*, 51(5), 698–717. https://doi.org/10.1080/00273171.2016.1215898

Nguyen, H. V., & Waller, N. G. (2022). Local minima and factor rotations in exploratory factor analysis. *Psychological Methods*. Advance online publication. https://doi.org/10.1037/met0000467