A tutorial for Bayesian Integrative Factor models

Mavis Liang

2024-10-10

Table of contents

Preface

This is the tutorial to guide statisticians to use Baysian integrative factor models.

```
1 + 1
```

```
[1] 2
```

1 Quick start

Step 1: Prepare the package and data

A small simulated data can be downloaded here:

RDS format

Step 2: Run BMSFA

Step 3: Post-processing

Step 4: Visualization

```
1 + 1
```

```
[1] 2
```

2 Preliminary

2.1 Factor models and multi-study setting

This chapter introduce the theory behind factor models

See Knuth (1984) for additional discussion of literate programming.

```
1 + 1
```

```
[1] 2
```

3 Case study: nutrition data

3.1 Loading and previewing the data

The data used in this section is from... This data is not publicly available. Please contact the authors of the original study for access.

```
load("./Data/dataLAT_projale2.rda")
```

The resulting object is a list of 6 data frames, each corresponding to a different study. Each data frame contains information about the nutritional intake of individuals, and the columns represent different nutrients. From Study 1 to Study 6, the number of individuals (N_s) are 1364, 1517, 2210, 5184, 2478, and 959, respectively, and the number of nutrients (P) are all 42. Let's take a look at the first few rows of the first data frame to get an idea of the data structure.

```
# Check how many studies in the list
length(X_s2)
```

```
[1] 6
```

```
# Dimension of each study
lapply(X_s2, dim)
```

```
[[1]]
[1] 1364   42

[[2]]
[1] 1517   42

[[3]]
[1] 2210   42

[[4]]
```

```
[1] 5184   42

[[5]]
[1] 2478   42

[[6]]
[1] 959  42
```

```r
X_s2[[1]][1:5, 1:5]
```

```
  Animal Protein (g) Vegetable Protein (g) Cholesterol (mg)  SCSFA MCSFA
1            28.9560               14.7440          256.761 0.2665 0.939
2            33.6675                8.9710          104.217 0.2180 0.520
3            70.0000               31.0635          207.902 0.9845 1.692
4            20.6700               13.8240          148.921 0.0625 0.239
5            15.4250               10.5550           65.060 0.0090 0.033
```

3.2 Data preprocessing

```r
count_na_and_negatives <- function(df) {
  # Count NA values
  na_count <- sum(is.na(df))
  # Count negative values
  negative_count <- sum(df < 0, na.rm = TRUE)

  # Print counts
  cat("Number of NAs:", na_count, "\n")
  cat("Number of negative values:", negative_count, "\n")
}
invisible(lapply(X_s2, count_na_and_negatives))
```

```
Number of NAs: 1344
Number of negative values: 0
Number of NAs: 1344
Number of negative values: 1
Number of NAs: 1344
Number of negative values: 0
Number of NAs: 1344
Number of negative values: 2
```

```
Number of NAs: 1344
Number of negative values: 1
Number of NAs: 1344
Number of negative values: 0
```

```r
process_study_data <- function(df) {
  # Remove rows where all values are NA
  cleaned_df <- df[!apply(df, 1, function(row) all(is.na(row))), , drop = FALSE]
  # Count remaining rows
  remaining_rows <- nrow(cleaned_df)
  # Print results for the study
  cat("Remaining rows:", remaining_rows, "\n")
  return(cleaned_df)
}

Y_list <- lapply(X_s2, process_study_data)
```

```
Remaining rows: 1332
Remaining rows: 1485
Remaining rows: 2178
Remaining rows: 5152
Remaining rows: 2446
Remaining rows: 927
```

```r
# Replace negative values with 0, then log(x+0.01) + 0.01
replace_negatives <- function(df) {
  # Replace negative values with 0
  df[df < 0] <- 0
  # Apply log transformation
  transformed_df <- log(df + 0.01)
  return(transformed_df)
}

Z_list <- lapply(Y_list, replace_negatives)
#saveRDS(Z_list, "./Data/nutrition_processed.rds")
```

4 Summary

In summary, this book has no content whatsoever.

```
1 + 1
```

```
[1] 2
```

References

Knuth, Donald E. 1984. “Literate Programming.” *Comput. J.* 27 (2): 97–111. https://doi.org/10.1093/comjnl/27.2.97.