Benjamin Rozonoyer

Braverman's 1st Algorithm for Grouping Parameters

This is an implementation of the algorithm of grouping parameters described in the first section of Braverman's article (124-126), and then to implement a modification to it (described below). The algorithm iteratively regroups the parameters among a set number of groups, and halts at a configuration that maximizes the sum of correlations of the parameters measured within each group.

Braverman's Algorithm

Input: File (in csv format) with parameter vectors.
Initialization: The parameters are randomly split into the desired number *k* of groups (this is parametrized in my program), with a random seed that allows the results to be reproduced. This was essential when testing whether the final groupings depended on the initial random split. Both the number of groups and the initial split of the parameters among them influenced the convergence of the algorithm (to a local or global maximum). Both the parameters and the groups are numbered – their ordering is set at initialization, and the algorithm loops over them every time in this set order.

Algorithm:

At each iteration of the algorithm, the value of the functional J_1 is computed in accordance with Braverman's formula (1):

(1)
$$J_1 = \sum_{x_i \in A_1} (x_i, f_1)^2 + \sum_{x_i \in A_2} (x_i, f_2)^2 + \ldots + \sum_{x_i \in A_k} (x_i, f_k)^2.$$

As we see from above, J_1 is computed as the sum of components corresponding to each group $A_1 \ldots A_k$. For each group A_k, the program builds a correlation matrix for its parameters. For each group's correlation matrix, it finds the eigenvector corresponding to the maximum eigenvalue, and constructs a factor *f* as a normalized linear combination of parameters A_k, using the elements of the eigenvector as weights.

After computing the value of the functional J_1 at the current iteration, the algorithm loops over every parameter (in a set order, which is determined at initialization), and checks whether moving this parameter to another group increases the value of J_1. Here is an example. We take parameter x_1, and compute the value of J_1 with x_1 in its current group G_1. Then we loop over the groups (in a set order), and see if transferring x_1 from its present group to the new group increases J_1. Suppose that transferring x_1 to G_2 doesn't increase J_1, but transferring x_1 to G_3 does. Then we move x_1 from G_1 to G_3, and go into a new iteration. For a given parameter, the algorithm transfers it to the first group it encounters that increases the value of J_1 (if such a group exists), even if there is another group after it for which the increase in J_1

would be greater. Once a transfer is made, the algorithm goes into a new iteration (repeating the steps described in this paragraph).

Once an iteration does not result in an increase to J_1, the algorithm stops – it has converged to a local or global maximum, depending on the parameters and their initial configuration. The final groupings are the extreme parameter groupings discussed in Braverman's article. Additionally, the program computes the factor for each of these final groups, and for each factor plots a histogram of its components (the observations, projected onto the group's factor).

Modification to Braverman's Algorithm

The modification which I implemented to Braverman's algorithm is in fact a simplification, resulting in faster running time. Instead of computing the functional's value as in formula (1) above, we compute it as the sum, for each group of parameters, of the sum of *n* greatest eigenvalues $\lambda_1 \ldots \lambda_n$ of that group's correlation matrix:

(1′) $J_1' = (\lambda_1 + \ldots + \lambda_n)_{group\_1} + (\lambda_1 + \ldots + \lambda_n)_{group\_2} + \ldots + (\lambda_1 + \ldots + \lambda_n)_{group\_k}$

The program accepts as a parameter the number *n* of greatest eigenvalues included in the sum of J_1'. I ran it for the greatest and the sum of the first two eigenvalues ($n = 1,2$). This simplification lets us avoid computing the factor and the scalar products of each parameter with the factor during iteration. Only after the algorithm has converged do we compute the actual factor, and visualize the observations as in the program for the original algorithm.

Testing

I tested Braverman's algorithm and its modification on a number of datasets representing sociological, technical and medical data from files used as examples for factor analysis in STATA. Here is what I did:

- Tested Braverman's algorithm and its modification, using 1 max eigenvalue and 2 max eigenvalues
- Used 5 different datasets
- Used different random seeds for initial random splits into different size groups – which sometimes affected the achieved maximum
- Ran algorithm with different number of groups
- Compared results with Principal Component Analysis – the results were usually easily interpretable and consistent with PCA
- Program includes histograms for visualization of results

- Timed Braverman's algorithm and its modification – the modified algorithm runs faster, although the actual runtime difference depends very much on the dataset

---

Program Structure

The modified program consists of 4 modules:

1. *algorithm.py* – the main module containing the iterative Algorithm class, as well as the main method in which to create an Algorithm object
2. *prepare_input.py* – this module contains methods to read in parameters from a file and split them randomly into the desired number of groups.
3. *group_calculations.py* – This module contains methods dealing with correlation matrices, eigenvalues, and matrix multiplication required over the course of the algorithm.
4. *visualization.py* – methods to plot histograms (very basic)

The input files are stored in the *Input* folder. I have provided a log file called *logfile* (the writes to it are currently commented out – please uncomment them to see the output in the log file). The comments in my program are meant to serve as additional instructions and clarifications on the usage of each of its methods.

References

Э. М. Браверман, “Методы экстремальной группировки параметров и задача выделения существенных факторов”, *Автомат. и телемех.*, 1970, № 1, 123–132

http://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=at&paperid=9614&option_lang=rus