Appendix: Computations and Storage Requirements

All the experiments were performed in a Dell Precision M4400 (processor Intel core 2 Duo at 2.26 GHz) running Linux openSUSE 12.1. We coded our approach in **C** and used the `.C` interface to R to have a friendly data input interface, see Pend and Leew (2002) for a good introduction. The analysis and graphics were done in R. To solve (10) efficiently via the Hungarian method we borrowed the function `solve_LSAP` from the library `clue` of R whose source code is written in **C**, see Hornik (2011). Thus allowing us to incorporate it into our own **C** code directly. The reported CPU times were measured using the function `system.time` of R.

One of the main criticisms of KL relabeling is its storage requirements. In our implementation, the classification probabilities were stored in a text file directly, at each iteration of the Gibbs sampler, and we worked with 16 digits of precision. Then our KL implementation loaded all the matrices of classification probabilities from this text file. With this approach the memory used to store one classification probability in a text file was 18 bytes (0. + 16 digits), then 2 bytes to print a space after each classification probability ("\t") and 1 byte to print in the next row of the text file ("\n"). Thus, the text file size to store N matrices of classification probabilities of dimension $n \times k$ was of

$$\text{TFScp}(N, n, k) = (18kn + 2kn + n)N = n(20k + 1)N \quad \text{bytes.} \tag{1}$$

On the other hand, the text file to store N allocation vectors of dimensions $1 \times N$ used by Data and ECR relabelings was

$$\text{TFSalloc}(N, n) = (3n + 1)N \quad \text{bytes.} \tag{2}$$

This follows since 1 byte is needed to store one allocation, 2 bytes to print a space after each allocation, and 1 byte to print in the next row of the text file for each allocation vector.

Then, with the precision considered, we see that the text file size to store the classification

probabilities is approximately $7k$ times larger than the one used to store the vectors of allocations. We can use (1) and (2) to work out the size of the text files used in our experiments. These are displayed in Table 1. Note that 1 MB = 1024^2 bytes and 1 GB = 1024^3 bytes.

Data Set	k	n	N	TFScp	TFSalloc
Model 1	2	1000	30,000	1.1 GB	85.9 MB
Model 2	4	200	30,000	463.5 MB	17.2 MB
Model 3	5	600	30,000	1.7 GB	51.5 MB

Table 1: Text file size for the classification probabilities and latent allocations.

The size of the text files shown in Table 1 are easily handled by modern day computers, but to test our every day laptop we increased N gradually running the algorithms with the data set from Model 3. As our laptop's processor is of 32 bytes, we experienced problems to store text files larger than 2 GB, however this was easily solved enabling the large file support in Linux, see Jaeger (2005). When we reached $N = 110,000$, and hence 6.2 GB of size for the classification probabilities text file, the KL algorithm crashed when loading all the matrices of classification probabilities. If the aim when performing component specific inference is classification analysis, then the N matrices of classification probabilities sampled throughout the MCMC are needed.

If the Data relabeling is used, we can store all the classification probabilities directly into a text file, as described before, and then calculate (5) reading just one matrix of classification probabilities at a time. In our experiments we handled text file sizes of classification probabilities of 15 GB of memory without any problem. Indeed, a possible solution for the KL strategy to handle larger text files could be to read one matrix of classification probabilities at a time and return the file pointer to the beginning of the text file, at the end of each iteration. However, this would slow the KL algorithm even more.

REFERENCES

Hornik, K. (2011), *clue: Cluster ensembles*, R package version 0.3-40, `http://CRAN.R-project.org/package=clue`,

Jaeger, A. (2005). "Large File Support in Linux", `http://www.suse.de/~aj/linux_lfs.html`.

Peng, R.D., and Leeuw, J. (2002), An Introduction to the `.C` Interface to R, UCLA: Academic Technology Services, Statistical Consulting Group, `http://www.ats.ucla.edu/stat/r/library/interface.pdf`