Approximating a similarity matrix by a latent class model

Cajo J.F. ter Braak, Yiannis Kourmpetis, Henk A. L. Kiers, Marco C. A. M. Bink

Approximating a similarity matrix by a latent class model

Cajo J.F. ter Braak[a1], Yiannis Kourmpetis[a], Henk A. L. Kiers[b], Marco C. A. M. Bink[a]

[a]Biometris, Wageningen University and Research Centre, Box 100, 6700 AC
Wageningen, the Netherlands,

[b]Heymans Institute of Psychology, University of Groningen, Grote Kruisstraat 2/1,
9712 TS Groningen, The Netherlands

[1] cajo.terbraak@wur.nl

Abstract

Let **Q** be an $n \times n$ square symmetric matrix **Q** of nonnegative elements between 0 and 1, *e.g.* similarities. In the paper we interpret elements of **Q** as the probability that two individuals are identical in some sense, *e.g.* belong to the same class, and are thus confused by a measuring device. We want to model this matrix by a latent class model with K classes with K fixed, but unknown. Members of the same class are always confused. Let **P** be an $n \times K$ matrix with assignment probabilities of individuals to classes ($0 \leq p_{ik} \leq 1$ and row sums of 1). Our problem then amounts to approximating **Q** by $\mathbf{PP}^T$, while disregarding the diagonal elements of **Q**. In a least-squares formulation, it is not an eigen problem because of the constraints on **P**. It is also more difficult than standard quadratic programming or fitting a latent budget model as our problem is quartic rather than quadratic in the parameters. We present two algorithms to solve the problem; a brute force genetic algorithm (differential evolution) and an iterative row-wise quadratic programming approach. Both algorithms converged to the same minimum on simulated data, which made it more likely that they reached the global minimum.

Key words: non-negative matrix decomposition, latent class model, network, fuzzy clustering, differential evolution

1. Introduction

The data in this paper forms an $n \times n$ square symmetric matrix **Q** of nonnegative elements between 0 and 1. The elements of **Q** could measure the similarities of individuals in a social network, of genes in a gene network or of products in market science. We will think about the elements of **Q** as the probability that two individuals are judged the same by some measuring device, or equivalently the probability that the device confuses the two individuals.

We wish to summarize the similarities among individuals by imposing a simple model. Three kinds of models are often used in data mining: distance models (Borg and Groenen 1997), vector models such as the bilinear model (Jolliffe 1986) and cluster analysis models including latent class models (Heinen 1996, McLachlan and Peel 2000). Each of these models has been applied to network data (Nowicki and

Snijders 2001, Hoff *et al.* 2002, Hoff 2005). Latent classes tend to be easier to communicate to the general public than vector models. Distance models take perhaps an intermediate position. The vector model would be much easier to understand if it had a class interpretation. In this paper we present a constrained vector model. The constraints are formulated so as to allow a latent class interpretation.

In this paper we propose approximating the off-diagonal elements of **Q** by $\mathbf{PP}^T$ with **P** an $n \times K$ matrix with non-negative elements and K a fixed value. In addition, the rows of **P** must sum to unity. Without the sum constraint, the model amounts to nonnegative matrix factorization (Lee and Seung 1999) for symmetric nonnegative matrices (Catral *et al.* 2004, Ding *et al.* 2005). With the sum constraint, the model can be viewed as a latent class model, as we show in this paper.

Table 1 shows an example matrix (**Q**) for six individuals named *A* to *F*. Individual *A* is never confused with any other individual. Individuals *B-D* are always confused. The individuals *E* and *F* are confused with probability 0.7. This matrix can arise when the individuals belong to four classes, labeled C_1- C_4 in Table 1 and members of the same class are always confused. Individual *A* belongs to a unique class C_1 and individuals *B-D* all belong to a single class C_2 in Table 1. The confusion probability of 0.7 between individuals *E* and *F* may arise when *E* is always a class C_3 type and *F* belongs to C_3 with probability 0.7 and thus with probability 0.3 to another class, named C_4 in Table 1. The solution is not unique. For instance, a confusion probability of 0.7 also arises with *E* of types C_3 and C_4 with probabilities 0.25 and 0.75 and *F* of types C_3 and C_4 with probabilities 0.1 and 0.9, respectively, as $0.25 \times 0.1 + 0.75 \times 0.9 = 0.7$. Also, solutions with more than four classes would also give a perfect fit. In the matrix **P** in Table 1, *F* may for instance belong to two classes C_4 and C_5 with probabilities summing to 0.3. For our purpose, all these solutions are equivalent as they yield the same **Q**. In general, the number of classes is unknown. We will seek the smallest number that gives a good fit of **Q**.

In section 2 we derive and characterize the model and the approximation problem that it gives. In the model each individual belongs with a certain probability to a particular class. We call it a latent class model for similarity matrices. We present two algorithms to solve it. Section 3 presents a brute force genetic algorithm (differential evolution) and section 4 an iterative row-wise quadratic programming approach. In section 5 we describe the performance of both algorithms on artificial data and we conclude with a summary of our findings.

2. The latent class model for a similarity matrix

Let **Q** be an $n \times n$ square symmetric matrix with elements q_{ij} between 0 and 1, *e.g.* denoting the confusion probability of individuals *i* and *j*. We seek a model that allows us to sample or draw class memberships for each of the *n* individuals in such a way that the probability that individual *i* and *j* are of the same class (*i.e.* are confused) is q_{ij} for all $i \neq j$ ($i = 1,\ldots, n$; $j = 1,\ldots, n$). The classes are unknown; also their number is unknown, but we hypothesize a size *K* from now on for some value of *K*.

We assume here the transitivity property: if individuals *i* and *j* are of the same type and individuals *i* and *k* are also of the same type in a draw from the model, then individuals *j* and *k* should be of the same type in this draw so that *i*, *j* and *k* are all of the same type, *i.e.* they fall in the same class in this draw. A simple model that fits our purpose is a model with K disjoint latent classes in which each individual belongs to precisely one latent class. We extend this model with probabilities. Let **P** be an $n \times K$ matrix with elements p_{ik} being the probability that individual i belongs to class k. Note that

$$0 \le p_{ik} \le 1 \text{ and } \sum_{k=1}^{K} p_{ik} = 1 \quad (i = 1,\ldots, n;\ k = 1,\ldots, K). \tag{1}$$

By drawing the class memberships for each individual i from the i^{th} row of **P** independently, the probability that individuals i and j fall in the same class (the coincidence probability) is

$$q_{ij}^{*} = \sum_{k=1}^{K} P(i \in class(k) \wedge j \in class(k)) = \sum_{k=1}^{K} p_{ik} p_{jk} \quad \forall i \ne j\ . \tag{2}$$

The problem that we wish to solve is to find a matrix **P** such that q_{ij}^{*} is close to the observed q_{ij} for all $i{\neq}j$. The $\{q_{ij}^{*}\}$ are thus the coincidence probabilities induced by a latent class model with memberships probabilities **P**.

For a mathematical convenience we use least-squares approximation. The problem then is to minimize the loss function

$$f(\mathbf{P}) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} (q_{ij} - \mathbf{p}_i^T \mathbf{p}_j)^2\ , \tag{3}$$

where $\mathbf{p}_i^T$ denotes the i^{th} row of **P,** subject to the nK non-negativity and n equality constraints in (1). We will report the loss in terms of the root mean squared error (RMSE), defined as $\sqrt{2f(\mathbf{P})/(n(n-1))}$.

We make the following observations.

(1) In matrix notation the problem can almost be written as the squared Frobenius norm $\left\|\mathbf{Q} - \mathbf{P}\mathbf{P}^T\right\|^2$ except that that the diagonal elements do not count. Without constraints (1) and with the diagonal counting, the optimal **P** could be derived from an eigen analysis of **Q**, which is an algorithm often used for principal components analysis. It is unclear, however, how to take advantage of this for solving the constrained problem.

(2) Our problem is reminiscent of latent budget analysis (Mooijaart *et al.* 1999) and archetypal analysis (Cutler and Breiman 1994), where a rectangular an $n \times m$ matrix **C** with non-negative elements that sum row-wise to 1 (each row being a composition) is approximated by $\mathbf{AB}^T$ with **A** and **B** being of size $n \times K$ and $m \times K$, respectively, having non-negative elements and each row **A** and each column of **B** summing to unity. So, if $n = m$, **C** is symmetric and **A** =**B** the two problems coincide. Mooijaart *et al.* (1999) estimate the latent budget analysis model by alternating least-squares. This is a very convenient method because solving for **A,** while keeping **B** fixed, is a constrained quadratic programming problem as is solving for **B,** while keeping **A** fixed. This alternating least-squares method is not available in our problem.

(3) Without the sum constraint, our problem is also akin to non-negative matrix factorization (Lee and Seung 1999) for which several algorithms have been proposed (Chu *et al.* 2004). Non-negative matrix factorization is in fact equal to latent budget analysis without sum constraints.

(4) Loss function (3) involves terms of the form $p_{ik}^2 p_{jk}^2$, $p_{ik} p_{jk} p_{il} p_{jl}$ and $p_{ik} p_{jk}$ ($i \neq j$); it is thus rather quartic than quadratic in the parameters, although terms of the form p_{ik}^4 do not enter the loss function because the diagonal of **Q** is excluded.

(5) The matrix **P** has $n(K$-1) free non-negative parameters and the data matrix **Q** has $n(n$-1)/2 off-diagonal elements. If the problem were linear and unconstrained, a perfect fit would be feasible for $n(K$-1)> $n(n$-1)/2, *i.e.* for K > (n+1)/2. This lower bound for K is not always attainable of course. For example, if **Q** is diagonal, then K = n gives a perfect fit and smaller values of K do not.

3. Differential evolution (DE) approach

Loss function (3) with constraints (1) is not convex and thus potentially contains many local minima, even beyond the minima that are generated by rearranging of the columns of **P**. This is the reason we attempted a global optimization method, in particular Differential Evolution (Storn and Price 1997, Price *et al.* 2005). Differential Evolution (DE) is a derivative-free global optimization method that belongs to the family of Evolutionary Algorithms. In DE a population of solution vectors $\mathbf{x}_1 \ldots \mathbf{x}_N$ is used to explore the solution space and is maintained for a number of generations till the population reaches the minimum. In our problem, each member vector $\mathbf{x}_i$ = vec($\mathbf{P}_i$) with $\mathbf{P}_i$ a trial solution of (3) subject to (1). The size of the population depends on the problem and the other parameters of the DE, but in general it should be higher than the number of parameters d. After some trials we chose N=2d = 2nK.

We initialized each members vector $\mathbf{x}_i$ (i = 1, … , N) independently with nK random values between 0 and 1 and then divided the K values of each of the n individuals by their sum so as to satisfy constraints (1). The exploration of the space is carried as follows in DE.

A new "mutant" vector $\mathbf{x}_i^*$ is proposed for each member vector $\mathbf{x}_i$ in turn, using three different randomly selected vectors of the population

$$\mathbf{x}_i^* = \mathbf{x}_{r_0} + F(\mathbf{x}_{r_1} - \mathbf{x}_{r_2}) \qquad (4)$$

with F a scalar. In addition to (4), another genetic operation is used in the DE, namely crossover. With a crossover rate CR (0<CR≤1), a fraction CR of the elements in $\mathbf{x}_i$ are mutated according to (4), whereas the remaining parameters are left unchanged, *i.e.* are set equal to the corresponding values in $\mathbf{x}_i$. We chose to apply the crossover operator on the level of the individuals within the parameter vector, *i.e.* the parameters of an individual are either mutated according to (4) or left unchanged. In terms of $\mathbf{P}_i$, each row of $\mathbf{P}_i$ is thus independently selected for mutation, the selection probability being CR. The resulting mutant vector $\mathbf{x}_i^*$ does not normally satisfy the constraints (1). We therefore replace any negative value in $\mathbf{x}_i^*$ by 0 and any value greater than 1

by 1 and then divide the *K* values of each of the *n* individuals within $\mathbf{x}_i^*$ by their sum. After these operations the mutant $\mathbf{x}_i^*$ satisfies the constraints. The member vector $\mathbf{x}_i$ is replaced by $\mathbf{x}_i^*$ if $f(\mathbf{x}_i^*) \leq f(\mathbf{x}_i)$ with *f*(.) the loss function. Each possible update of $\mathbf{x}_i$ requires one function evaluation, when loss function values of members are stored.

The parameter *F* determines the step size. In a Markov Chain Monte Carlo (MCMC) version of DE, ter Braak (2006) argued that *F* should be proportional to $d^{-0.5}$ so as to make the step length $\left\|\mathbf{x}_i^* - \mathbf{x}_{r_0}\right\|$ approximately dimension independent. In a further investigation of this, Price and Rönkkönen (Price and Rönkkönen 2006) distinguished between Global Search (GS) schemes where $r_0 \neq i$ and Local Search (LS) schemes where $r_0 = i$ and derived optimal dimension dependent functions for *F* for each scheme from experiments with simple loss functions. Figure 1 shows the two functions. In GS, the optimal *F* decreases very slowly with *d* ($F_1 = 0.748d^{-0.1206}$), whereas in LS the square-root rule holds ($F_2 = \sqrt{2/d}$). In the MCMC, LS is used albeit with $F = \sqrt{2.83/d}$ (ter Braak 2006). We let *d* depend on the real number of elements updated in a crossover, the mean dimension with crossover rate CR being $\overline{d} = CR \times n \times K$ (Figure 1). We were interested in the performance of the four scenarios obtained by crossing GS versus LS with F_1 versus F_2. We also tried three values of CR, 0.1, 0.5 and 0.9 within each scenario, yielding 12 runs for each **Q** and *K*.

4. Iterative row-wise quadratic programming (QP$_{\text{irw}}$)

In this section we propose to use a row-wise iterative algorithm to minimize loss function *f*(**P**) in equation (3) subject to the constraints in (1). The advantage of this approach is that it leads for each row to a standard quadratic program.

To update $\mathbf{p}_i$, we minimize *f*(**P**) over $\mathbf{p}_i$, while keeping the other rows of **P** fixed. This boils down to minimizing

$$\| \mathbf{q}_i - \mathbf{P}_{-i}\mathbf{p}_i \|^2 \quad (5)$$

where $\mathbf{q}_i$ denotes the i^{th} column of **Q** without q_{ii}, and $\mathbf{P}_{-i}$ denotes matrix **P** after deleting row *i*, subject to the constraints

$$\mathbf{p}_i \geq \mathbf{0} \text{ and } \mathbf{p}_i^T\mathbf{1} = \mathbf{1} \quad (6)$$

where **0** and **1** denote vectors of appropriate lengths with all zero and unit elements, respectively.

The constraint $\mathbf{p}_i^T\mathbf{1} = \mathbf{1}$ can be enforced by reparametrization, as follows. We can always write

$\mathbf{p}_i = \mathbf{Uv} + \alpha\mathbf{1}$

with **U** a columwise orthonormal basis for the orthocomplement of **1**, and **v** and α a vector and scalar respectively. Because **U** and **1** jointly span the whole space, such vector and scalar always exist. Now the constraint that $\mathbf{p}_i^T\mathbf{1} = \mathbf{1}$ implies that $\mathbf{vU}^T\mathbf{1} + \alpha\mathbf{1}^T\mathbf{1} = 1$, and because $\mathbf{U}^T\mathbf{1}=\mathbf{0}$ and $\mathbf{1}^T\mathbf{1}=K$, it follows that $\alpha= K^{-1}$. Hence, we can always write

$\mathbf{p}_i = \mathbf{Uv} + K^{-1}\mathbf{1}$

and, as is readily verified, $\mathbf{p}_i$ thus specified, satisfies the constraint $\mathbf{p}_i^T\mathbf{1} = \mathbf{1}$ for every **v**. Hence, by reparameterizing $\mathbf{p}_i$ as above, it remains to minimize the function

$g(\mathbf{v}) = || \mathbf{q}_i - \mathbf{P}_{-i}K^{-1}\mathbf{1} - \mathbf{P}_{-i}\mathbf{Uv} ||^2$

over **v** subject to the constraint that $\mathbf{Uv} + K^{-1}\mathbf{1} \geq \mathbf{0}$. This problem can be recognized as a standard quadratic program or, equivalently, the so-called LSI problem solved by Lawson and Hanson (1995) of minimizing

$h(\mathbf{v}) = || \mathbf{f} - \mathbf{Ev} ||^2$

subject to $\mathbf{Gv} \geq \mathbf{h}$,

where

$\mathbf{f} = \mathbf{q}_i - K^{-1}\mathbf{P}_{-i}\mathbf{1}$ (where $\mathbf{P}_{-i}\mathbf{1} = \mathbf{1}$, due to the constraint $\mathbf{P}_{-i}\mathbf{1} = \mathbf{1}$)
$\mathbf{E} = \mathbf{P}_{-i}\mathbf{U}$
$\mathbf{G} = \mathbf{U}$
$\mathbf{h} = -K^{-1}\mathbf{1}$.

Lawson and Hanson (1995) provide a quick and effective algorithm for minimizing this function. After having thus found the optimal **v**, the vector $\mathbf{p}_i$ that minimizes f over $\mathbf{p}_i$ is given by $\mathbf{Uv} + K^{-1}\mathbf{1}$.

By iteratively updating each row of **P** as described above, repeating this cycle of sequential updates as long as the overall function value has not converged, eventually, the algorithm will converge to a stable function value, which hopefully is at least a local minimum.

We ran this algorithm from 10 independent random initial matrices **P** (as in DE) and observed the number of iterations (where one iteration consists of updating each row of **P** once).

5. Examples

In the case of the example **Q** of Table 1, Differential evolution (DE) and iterative row-wise quadratic programming (QP_{irw}) were both able to find a perfect fitting **P** with four classes. QP_{irw} required between 5 and 10 iterations and was much quicker than DE. For sake of completeness, the root mean squared error (RMSE) values for the best solution with two and three classes were 0.284 and 0.043.

Table 2 shows another 6×6 example of **Q**. The minimum RMSE values that we found were 0.254, 0.046, 0.022, 0.021and 0.021 for 2-6 classes, respectively. Neither DE nor QP_{irw} was able to find a perfect fitting **P**, not even with 6 classes. This strongly suggests that no such perfect solution exists. Table 2 shows the solution with 4 classes. The classes C_1 and C_3 express the similarity between individual *A* and *B* and *D*, *E* and *F*. Class C_2 expresses the uniqueness of individual C and class C_4 is needed to fit **Q** in more detail. The solution for $K = 5$, essentially splits class C_4 in two, yielding a slightly better fit to element q_{DF}.

In order to show that our algorithms work for larger **Q**, we simulated four matrices of order 20×20 in 2×2 layout of number of classes (5 or 10) and two methods of generating the values of **P** (one yielding a highly structured **P** and another yielding an ill-structured **P**). In the first method, the rows were first divided in classes of equal size and the rows falling in the class *k* were sampled from Dirichlet distribution with parameter $\boldsymbol{\alpha}_k$. The k^{th} element of $\boldsymbol{\alpha}_k$ was assigned the value 8 and the remaining elements were set to 2/(*K*-1). In the second method all rows are generated from a single Dirichlet distribution with parameter $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_K)^T$ with $\alpha_j \sim$ Uniform(0,1). Figures 2 and 3 display example **Q** and **P**, respectively.

For each matrix a perfect fit is found when *K* is greater than or equal to the true number of classes (Figure 4). QP_{irw} required *ca.* 5 second on a 3MHz desktop computer for carrying out the fit for $n = 10$, $K=10$ and 10 restarts. The random restarts often resulted in the same RMSE. Each fit required between 50 and 200 iterations when the number of classes was not greater than the true number. For larger number of classes the number of iterations went up to a maximum of *ca.* 1000 while using a convergence criterion for $f(\mathbf{P})$ of 10^{-6}. DE required *ca.* five times longer to converge. It often found a similar but never lower RMSE and did not always find the prefect fit. Figure 5 displays a trace plot of the highly structured case for $K = 10$. GS with F_1 appears to converge best, even with the slow start with CR = 0.1. The square-root rule performed worse than F_1, both in GS and LS. . In the LS the square-root rule tends to converge prematurely to local minimum, in particular with high CR or high *d*. GS with F_1 and CR = 0.9 did best. It reached a perfect fit for the true *K* for all four **Q** matrices.

6. Discussion

This paper proposes a latent class model for approximating a similarity matrix and presents two algorithms for fitting the model. We first discuss our model and then the algorithms.

The key identity is equation (2), which gives the coincidence probability of two individuals, *i.e.* the probability that they fall in the same class. In our approach the similarity matrix is approximated by the coincidence probabilities induced by a latent class model. There are several related approaches.

Fuzzy clustering uses memberships that satisfy the constraints in equation (1) and is thus in spirit similar to our aims. Fuzzy c-means (Bezdek 1981) works with rectangular data. Hathaway *et al.* (1989) developed a fuzzy *c*-means algorithm that works on relational data. In their approach the similarity matrix is implicitly transformed to an underlying coordinate space where fuzzy *c*-means could be applied. Fuzzy *c*-means is least-squares on squared distances between individuals and cluster means and this property is extended to the kernel version by the 'kernel trick'(Hathaway 2005). By contrast, our approach is directly least-squares in terms of the similarity matrix. Ding *et al.*(2005) explore the relationship between kernel *c*-means and symmetric non-negative matrix factorization.

In proximity-based fuzzy clustering (Pedrycz *et al.* 2004), equation (2) is replaced by

$$q_{ij}^{*} = \sum_{k=1}^{K} (p_{ik} \wedge p_{jk}) \quad \forall i \neq j ,$$

where $\wedge$ denotes the minimum operation. This equation is based on fuzzy calculus whereas ours is firmly based on probability. Pedrycz et al. (2004) use a weighted least-squares approach. Their model does not require transitivity. While developing our model we had examples in mind in which transitivity holds true. Overlapping clustering (Arabie *et al.* 1981) differs in that memberships are 0 or 1 and do not need to sum to 1 per individual.

In the introduction we showed by example that the model has no unique solution for n=K=2. Nevertheless the two algorithms often found the same **P**. Uniqueness conditions are clearly an area of further research. The network properties of the model also deserve further study.

We developed two algorithms for fitting our model. Our DE approach was developed before we realized that the problem could be formulated as a sequential quadratic programme. DE helped to convince us that QP$_{\text{irw}}$ really worked, also when no perfect fit was achievable. Our experiments with DE confirm the conclusion of Price and Rönkkönen (2006) that global search needs large *F* value (F $\geq$ 0.4) and high CR speeds up convergence (if the minimum can be reached with high CR). With more insight in the problem, we developed QP$_{\text{irw}}$ which beats DE in speed. Further research should be point out whether QP$_{\text{irw}}$ is efficiently enough to speedily analyze problems with large *K* and *n*. QP$_{\text{irw}}$ is similar to the projected Newton method in Chu *et al.* (2004) to factorize non-negative matrices. We may also wish to replace the least-squares loss function by an entropy-based function (Lee and Seung 2001).

The model shares many features of already popular models. Yet it is different. Although we did not present a real application we expect the model to find many interesting applications in the near future.

Acknowledgements

We thank Patrick Groenen and Martin Boer for helpful discussion.

References

Arabie, P., Carrol, J. D., DeSarbo, W. and Wind, J., 1981. Overlapping clustering: A new method for product positioning. J. of Marketing Research, 18, 310-317.

Bezdek, J. C., 1981. Pattern recognition with fuzzy objective function algorithms. Plenum, New York.

Borg, I. and Groenen, P., 1997. Modern multidimensional scaling theory and applications. Springer, New York.

Catral, M., Han, L., Neumann, M. and Plemmons, R. J., 2004. On reduced rank nonnegative matrix factorization for symmetric nonnegative matrices. Linear Algebra and its Applications, 393, 107-126.

Chu, M., Diele, F., Plemmons, R. J. and Ragni, S., 2004. Optimality, computation, and interpretation of nonnegative matrix factorization.

Cutler, A. and Breiman, L., 1994. Archetypal analysis. Technometrics, 36.

Ding, C., He, X. and Simon, H. D., 2005. On the equivalence of nonnegative matrix factorization and spectral clustering. Proc. SIAM Int'l Conf. Data Mining, 606-610.

Hathaway, R. J., Davenport, J. W. and Bezdek, J. C., 1989. Relational duals of the c-means clustering algorithms. Pattern Recognition, 22, 205-212.

Hathaway, R. J. R. J., 2005. Kernelized non-euclidean relational c-means algorithms. Neural, parallel & scientific computations, 13, 305-326.

Heinen, T., 1996. Latent class and discrete latent trait models. Similarities and differences. Sage, London.

Hoff, P. D., 2005. Bilinear mixed-effects models for dyadic data. Journal of the American Statistical Association, 100, 286-295.

Hoff, P. D., Raftery, A. and Handcock, M. S., 2002. Latent space approaches to social network analysis. Journal of the American Statistical Association, 97, 1090-1098.

Jolliffe, I. T., 1986. Principal component analysis. Springer Verlag, New York.

Lawson, C. L. and Hanson, R. J., 1974. Solving least squares problems. Prentice-Hall, Englewood Cliffs, N.J.

Lee, D. D. and Seung, H. S., 1999. Learning the parts of objects by non-negative matrix factorization. Nature, 401, 788-791.

Lee, D. D. and Seung, H. S., 2001. Algorithms for non-negative matrix factorization. In: T. G. Dietterich and V. Tresp (Ed.^Eds), Advances in neural information processing, Vol. 13, MIT Press, 556-562.

McLachlan, G. and Peel, D., 2000. Finite mixture models. Wiley, New York.

Mooijaart, A., van der Heijden, P. G. M. and van der Ark, L. A., 1999. A least squares algorithm for a mixture model for compositional data. Computational Statistics & Data Analysis, 30, 359-379.

Nowicki, K. and Snijders, T. A. B., 2001. Estimation and prediction for stochastic blockstructures. . Journal of the American Statistical Association, 96, 1077-1087.

Pedrycz, W., Loia, V. and Senatore, S., 2004. P-fcm: A proximity-based fuzzy clustering. Fuzzy Sets and Systems, 148, 21-41.

Price, K. V. and Rönkkönen, J. 2006. Comparing the uni-modal scaling performance of global and local selection in a mutation-only differential algorithm. IEEE

Congress on Evolutionary Computation, CEC 2006, Vancouver, Canada, IEEE pp. 2034-2041.

Price, K. V., Storn, R. M. and Lampinen, J. A., 2005. Differential evolution, a practical approach to global optimization. Springer, Berlin.

Storn, R. and Price, K., 1997. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11, 341 - 359.

ter Braak, C. J. F., 2006. A markov chain monte carlo version of the genetic algorithm differential evolution: Easy bayesian computing for real parameter spaces. Statistics and Computing, 16, 239-249.

Table 1. Artificial 6 × 6 **Q** matrix for six individuals labeled *A-F* and a 6 × 4 matrix **P** with classes labeled C_1 - C_4, giving a perfect fit to the off-diagonal elements of **Q** by the formula $\mathbf{PP}^T$.

$$\mathbf{Q}=\begin{array}{ccccccc} & A & B & C & D & E & F \\ A & 1 & 0 & 0 & 0 & 0 & 0 \\ B & 0 & 1 & 1 & 1 & 0 & 0 \\ C & 0 & 1 & 1 & 1 & 0 & 0 \\ D & 0 & 1 & 1 & 1 & 0 & 0 \\ E & 0 & 0 & 0 & 0 & 1 & 0.7 \\ F & 0 & 0 & 0 & 0 & 0.7 & 1 \end{array} \qquad \mathbf{P}=\begin{array}{ccccc} & C_1 & C_2 & C_3 & C_4 \\ A & 1 & 0 & 0 & 0 \\ B & 0 & 1 & 0 & 0 \\ C & 0 & 1 & 0 & 0 \\ D & 0 & 1 & 0 & 0 \\ E & 0 & 0 & 1 & 0 \\ F & 0 & 0 & 0.7 & 0.3 \end{array}$$

Table 2. Artificial 6 × 6 **Q** matrix for six individuals labeled *A-F* and the best fitting 6 × 4 matrix **P** with classes labeled C_1 - C_4, giving RMSE = 0.022.

$$\mathbf{Q}=\begin{array}{ccccccc} & A & B & C & D & E & F \\ A & 1 & 0.9 & 0.2 & 0 & 0.1 & 0 \\ B & 0.9 & 1 & 0.1 & 0 & 0 & 0 \\ C & 0.2 & 0.1 & 1 & 1 & 0 & 0 \\ D & 0 & 0 & 0 & 1 & 0.8 & 0.7 \\ E & 0.1 & 0 & 0 & 0.8 & 1 & 0.9 \\ F & 0 & 0 & 0 & 0.7 & 0.9 & 1 \end{array} \qquad \mathbf{P}=\begin{array}{ccccc} & C_1 & C_2 & C_3 & C_4 \\ A & 0.88 & 0.09 & 0.03 & 0 \\ B & 1 & 0 & 0 & 0 \\ C & 0.12 & 0.88 & 0 & 0 \\ D & 0 & 0 & 0.79 & 0.21 \\ E & 0.02 & 0 & 0.98 & 0 \\ F & 0 & 0 & 0.90 & 0.10 \end{array}$$

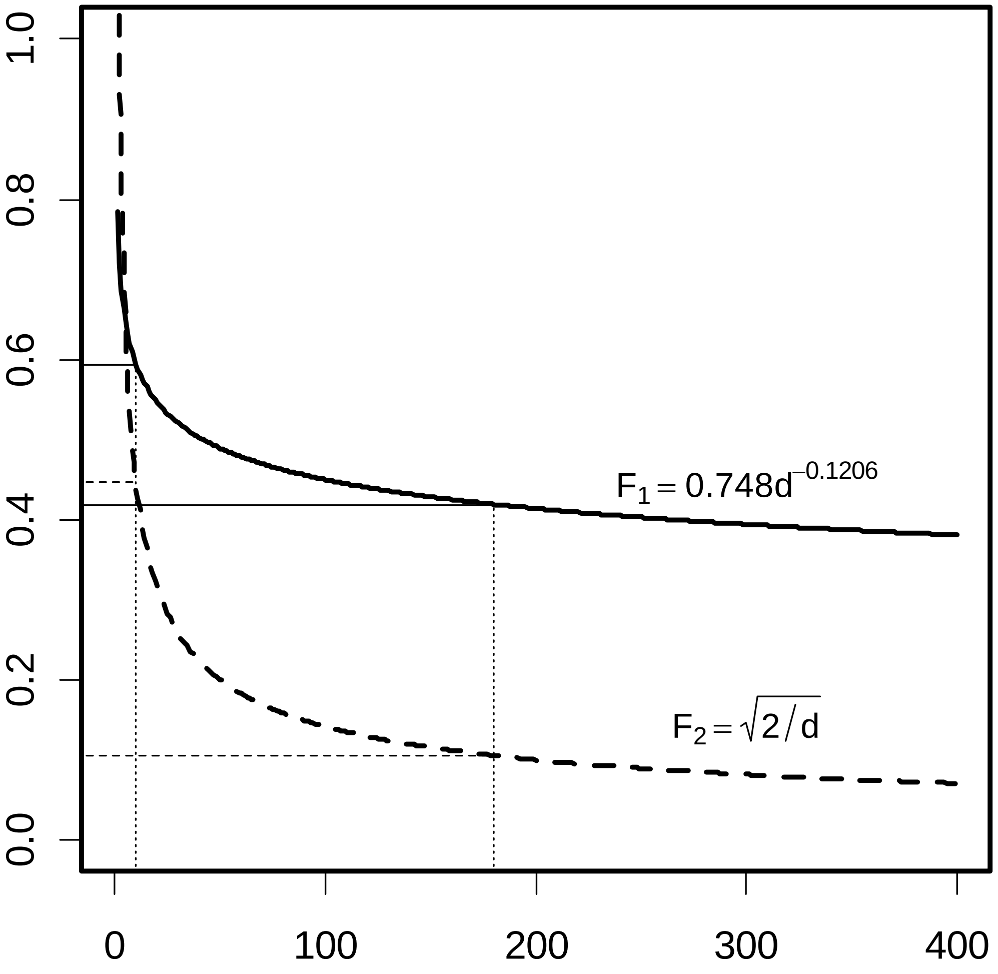

Figure 1. *F* in relation to dimension *d* = *nK* for two functions. Vertical lines at *d* = 10 and 180, corresponding to the mean number of parameters updated per function evaluation for n = 20 with CR = 0.1, *K* = 5, *n* = 20, and CR = 0.9, *K* = 10, *n* = 20, respectively.

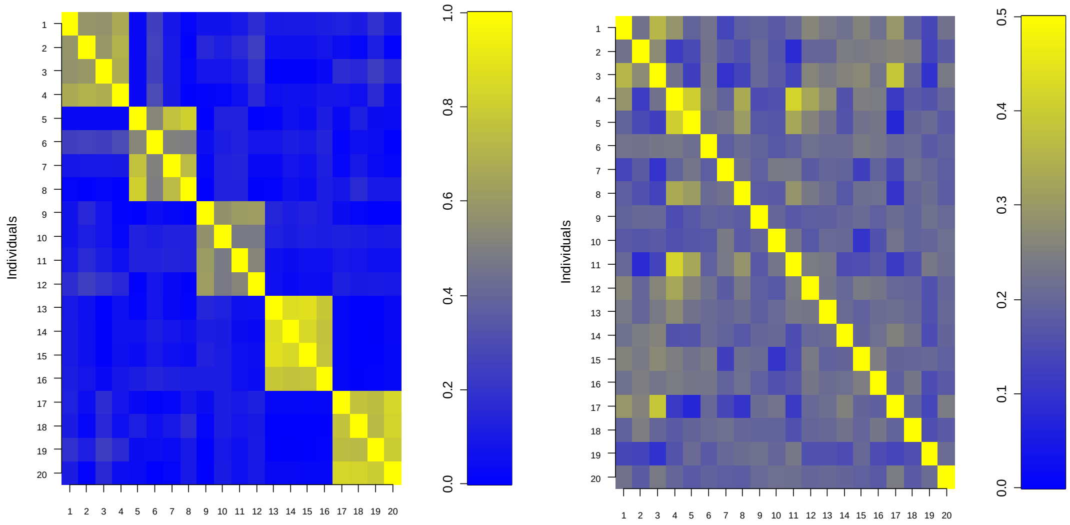

Figure 2. Example 20×20 **Q** matrix with *K* = 5 (left: highly structured; right: ill-structured).

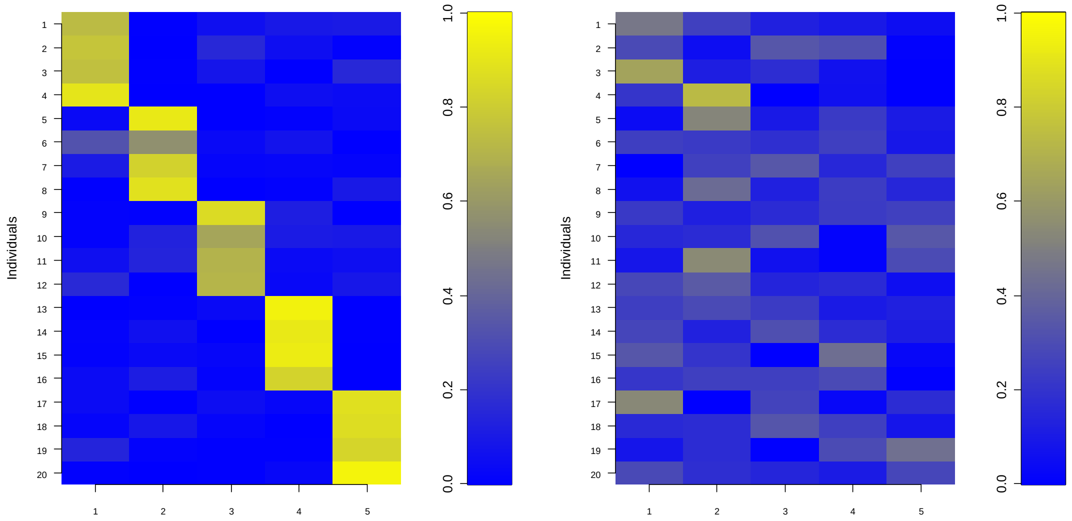

Figure 3. Estimated, perfectly fitting **P** matrices corresponding to the **Q** matrices of Figure 2.

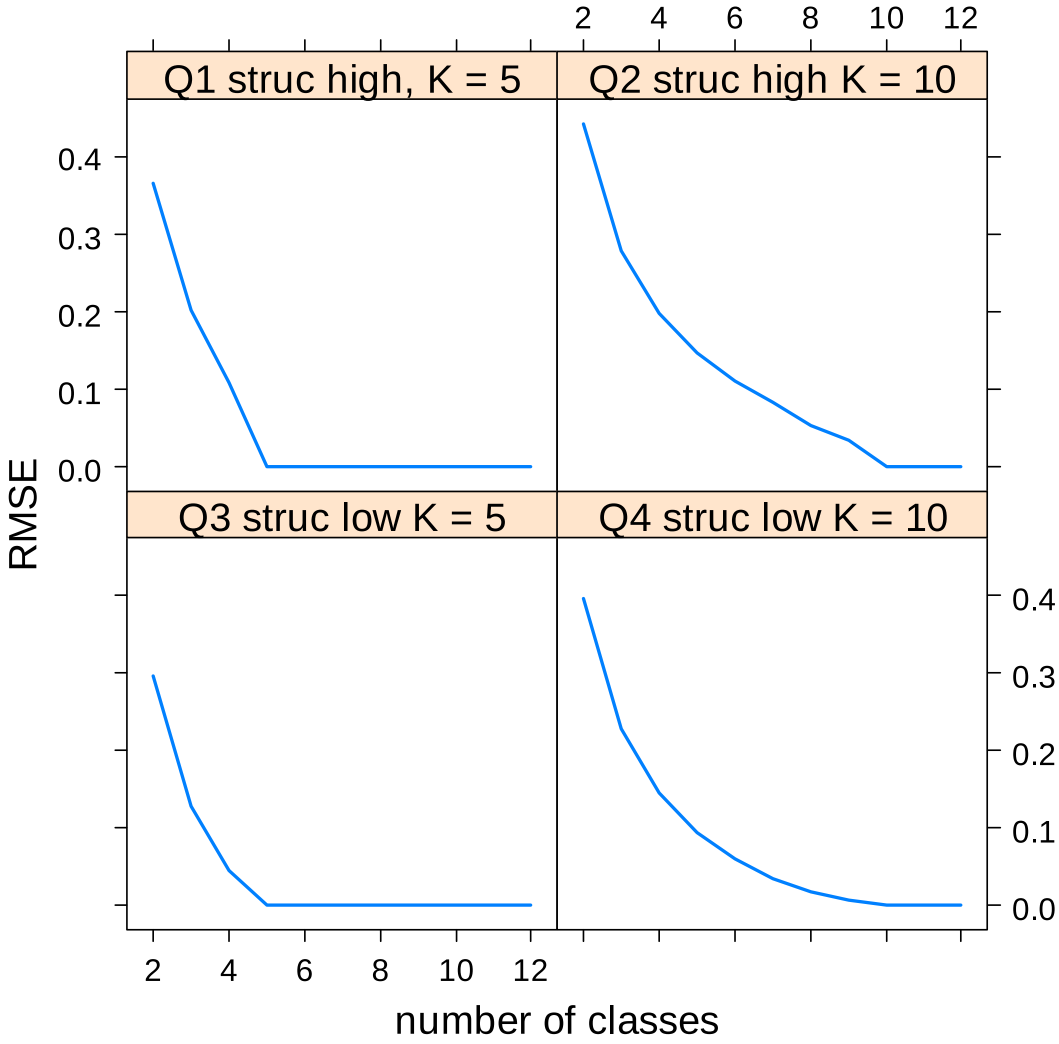

Figure 4. Scree plot for the four simulated 20×20 **Q** matrices (top row: highly structured **Q**, bottom row: ill-structured **Q**; left column: true *K* = 5; right column: true *K* = 10).

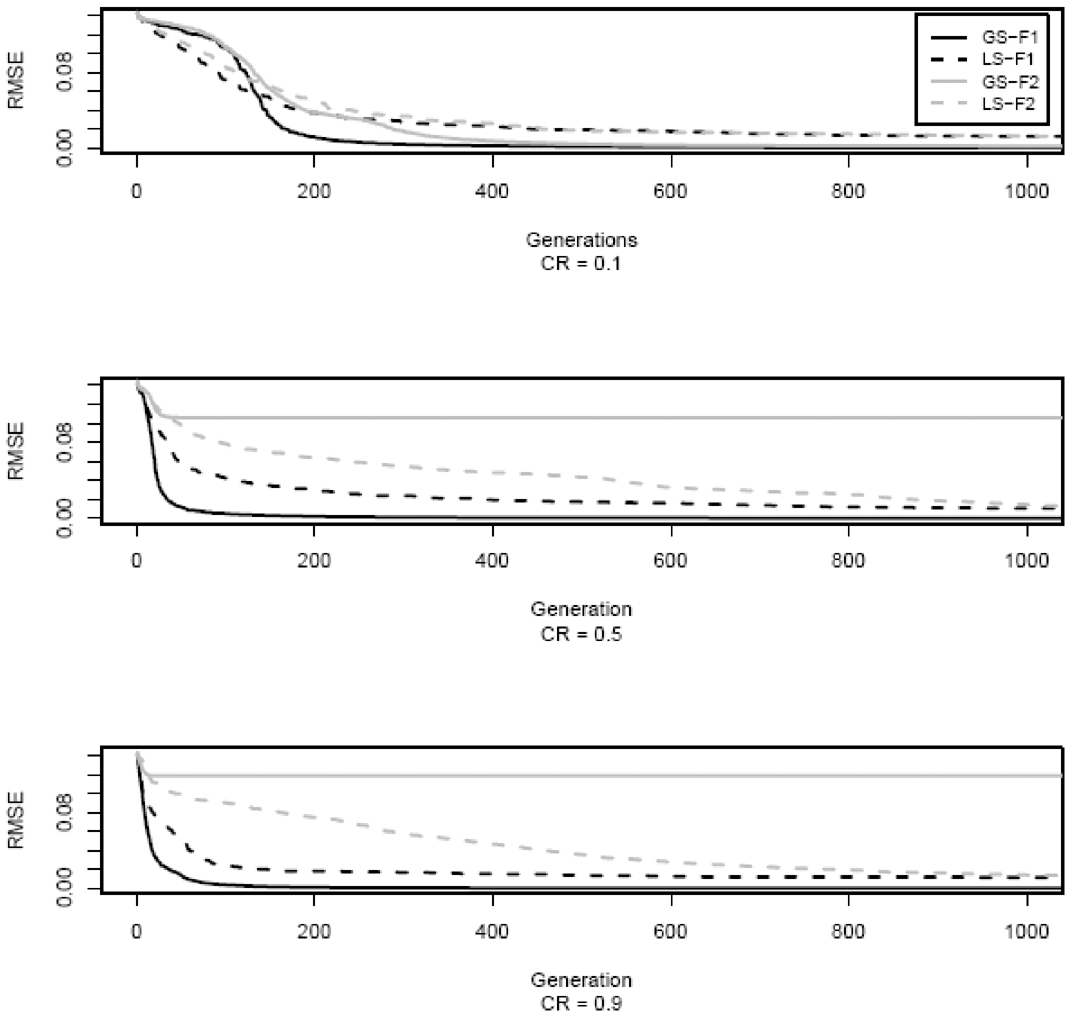

Figure 5. Trace plot for the highly structured **Q** with 10 true classes for *K* = 10 for the four scenarios of DE with CR = 0.1 (top), 0.5 (middle) and 0.9 (bottom).