Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 3, No. 2, pp. 387 – 397, May 2007

OPTIMIZATION SOLVERS AND PROBLEM FORMULATIONS FOR SOLVING DATA CLUSTERING PROBLEMS

Julien Ugon

Abstract: A popular approach for solving complex optimization problems is through relaxation: some constraints are removed in order to have a convex problem approximating the original problem. On the other hand, direct approaches for solving such problems are becoming increasingly powerful. This paper examines two cases drawn from data analysis, in order to compare the two techniques.

Key words: *global optimization, nonconvex, relaxation*

Mathematics Subject Classification: *49M20, 49M37, 65K05, 90C26, 90C30, 90C56*

1 Introduction

The field of mathematical optimization has many practical applications. Such applications generally consist of setting a number of factors in order to improve a process or to obtain the "best" outcome. The major difficulty of modeling this process into a mathematical optimization problem is that this problem should be accurate enough, while still solvable within a reasonable time.

Often, a same problem can have several mathematical formulations. A common practice is to try to reformulate it as a linear programming problem, for which commercial solvers such as CPLEX or LINGO provide efficient algorithms.

In many areas, however, these problems give rise to difficult optimization problems, and it is necessary to resort to more general software of global optimization. Commercial global solvers can be found in [3, 11, 14]. Due to the limitations of computing capabilities, it may be necessary to simplify a given problem, in order to get an approximate solution. For example, it may be beneficial to relax certain constraints to obtain a convex problem, much easier to solve. However, such an approach may lead to a solution with is not feasible for the original problem.

There exist surveys comparing various solvers for global optimization (see [9]). However, it is difficult to evaluate the efficiency of a solver, as problems in global optimization can differ very much, according to the form of the feasible set and of the objective function.

In this paper we consider two problems arising in real-world applications. For each of these problems, after providing a small descriptive background, several equivalent mathematical formulations are given and discussed. Then several methods, including relaxation techniques and commercial solvers are applied on these problems, and the results are compared and discussed.

2 First Problem

2.1 Setting of the Problem

The first problem under consideration arises in data analysis. Consider a dataset D with n_o observations and n_f features. We want to provide a method for clustering this dataset, by finding a hyperplane separating D in two clusters. We assume that this hyperplane passes through the barycenter of the set. To make sure that these two clusters are far apart, this hyperplane should as far as possible from any (that is the closest) point.

Consider the following problem (suggested in [8]):

$$\begin{array}{c} \text{maximize } f(a) = \min_{x\in D} |\langle a, x\rangle| \\ \text{subject to} \\ \|a\| = 1 \end{array} \tag{2.1}$$

where $\langle x, y\rangle$ is the scalar product over R^{n_f} and $\|\dot{\|} = \|\dot{\|}_2$ is the Euclidean norm. The number of variables is n_f. This problem is nonsmooth, nonconvex, therefore it is very difficult to solve. One of the main difficulties resides in the equality constraint. It is possible, however, to formulate several different problems leading to the solution of problem (2.1). In particular, the form of the objective function can be used in that purpose.

One possible formulation is to relax the equality constraint as follows

$$\begin{array}{c} \text{maximize } f(a) \\ \text{subject to} \\ \|a\| \leq 1 \end{array} \tag{2.2}$$

Another equivalent problem has the form:

$$\begin{array}{c} \text{maximize } \frac{1}{\|a\|} f(a) \\ \text{subject to} \\ a \in \mathrm{R}^{n_f} \setminus \{0\} \end{array} \tag{2.3}$$

Finally ([8]),

$$\begin{array}{c} \text{maximize } \min_{x\in D} \langle a, x\rangle^2 \\ \text{subject to} \\ \|a\| = 1 \end{array} \tag{2.4}$$

Each of these problems has advantages and disadvantages: in the case of problem (2.2), the constraint is quite simple, and the feasible set is closed convex. Problem (2.3) is a problem with a different type of constraint (over which unconstrained algorithms can be applied), and the objective function is constant along rays. However, this function is not defined at 0. Finally, problem (2.4) can be rewritten as a semidefinite problem which can be relaxed to a convex problem ([8]).

Consider the matrix $A = a^T a$. Then the problem (2.4) is equivalent to the following:

$$\begin{array}{c} \text{maximize } F(A) = \min_{x\in D} x^T A x \\ \text{subject to} \\ \text{trace}(A) = 1; \\ \text{rank}(A) = 1; \\ \text{A is symmetric positive definite.} \end{array} \tag{2.5}$$

It is possible to relax the rank constraint, in order to obtain the following convex problem:

$$\begin{gathered} \text{maximize } F(A) \\ \text{subject to} \\ \text{trace}(A) = 1; \\ \text{A is symmetric positive definite,} \end{gathered} \tag{2.6}$$

which can be rewritten as the semidefinite programming problem:

$$\begin{gathered} \text{maximize } c \\ \text{subject to} \\ c - x^T A x + y_x = 0, \forall x \in D; \\ \text{trace}(A) = 1; \\ y_x \geq 0, \forall x \in D; \\ \text{A is symmetric positive definite.} \end{gathered} \tag{2.7}$$

The problem (2.7) contains many more variables than the original problem: the number of variables is $n_f^2 + n_o + 1$. However, it is a linear problem on the cone of semidefinite matrices.

We want to compare two methods for finding a solution to the original problem (2.1) by solving reformulations of it:

Solving this problem directly using a global optimization solver.

Solving the relaxed problem (2.7), and try to find the solution of the original problem from the solution of the relaxed problem.

2.2 Numerical Experiments

In order to solve the semidefinite problem (2.7), the solver used is SeDuMi (see [13]). The solution A_* of the problem (2.7) is not directly applicable to the problem (2.1). As solutions of this problem we consider:

The leading eigenvector (that is the eigenvector corresponding to the largest eigenvalue) a^0 of A_*.

Random generation of 1000 vectors with correlation matrix A_*.

To solve the global optimization problem (2.2) and (2.3), we applied the following methods or software:

CIAO-GO (see [14])

AGOP (see [4, 15])

LGO (see [11])

Random generation of 1000 vectors according to normal distribution.

Problem (2.3) is more complex to solve, as its objective function is not defined at the origin.

Experiments are carried out on 8 small and medium size real world datasets. These datasets are (See [6, 7]):

aust Australian Credit Database;

breast Breast Cancer Database;

cleve Cleveland Database;

diab Diabetes Database;

firis Fisher's iris Database;

iono Ionosphere Database;

segment Image Segmentation Database;

vehic Vehicles Database;

wpbc Wisconsin Prognosis Breast Cancer Database

Various configurations have been examined: small or large number of observations, small or large number of features. SeDuMi is a C program with a Matlab interface. Unfortunately this interface limits the size of the datasets we can examine, and therefore no experiment has been carried out on large datasets.

The statement of problem (2.1) necessitates the hyperplane to go through the origin. Therefore the datasets were modified to have their barycenter at the origin.

2.3 Results of Numerical Experiments

Numerical experiments are presented in tables 1-5. The entries in these tables are as follows:

n_o Number of observations in the dataset;

n_f Number of features in the dataset;

A Solution of (2.6) obtained by SeDuMi;

$e(A)$ Leading eigenvector of matrix A;

r_A Random point with correlation matrix A;

Prob(i) Solution obtained for formulation (i);

T(i) Computational time for solving formulation (i);

N.F.(i) Number of Function Evaluations during solution of formulation (i).

2.3.1 SeDuMi

Table 1 presents the results obtained by the SeDuMi software for solving formulation (2.7).

Matlab is much slower than other packages and programs used during this research, and therefore the generation of random points can be very long. This generation was not taken into account in the results presented here. Only the time taken by SeDuMi to solve the formulation (2.6) is indicated.

Dataset	$F(A)$	$f(e(A))$	$\max(f(r_A))$	Time
aust	3350.01	0.004493	16.5428	143.0
breast	1.69996	0.00178	0.322707	37.9
cleve	40.6621	0.00505	0.998068	9.1
diab	78.4567	0.01393	1.10646	91.4
firis	0.6287	0.000288	0.122785	0.8
iono	0.1568	0.001619	0.0675656	31.5
segment	179.6	1.24×10^{-5}	0.686752	2723.0
vehic	302.7	0.006815	2.81461	158.0
wpbc	1076.2	0.001135	13.9543	34.0

Table 1: Results of numerical experiments for SeDuMi

2.3.2 CIAO-GO

CIAO-GO could only be applied on formulation (2.2). Formulation (2.3) could not be solved using CIAO-GO: the function not being defined at the origin lead to erroneous results due to computational inaccuracies. Results for CIAO-GO are presented in table 2. These results are quite good (much better than the results obtained through the relaxation technique), and found within a reasonable time.

Dataset	Prob. (2.2)	N.F.(2.2)	T(2.2)
aust	19.77	41402	0.938
breast	0.39631	26671	1.188
cleve	1.31029	32759	0.656
diab	1.65667	17336	0.609
firis	0.153028	6839	0.062
iono	0.0945898	153419	9.719
segment	0.564424	129257	11.859
vehic	1.76744	36256	1.391
wpbc	15.3779	76682	1.719

Table 2: Results obtained by CIAO-GO

2.3.3 AGOP

AGOP could be applied on both formulations 2.2 and (2.3). Results for AGOP are presented in table 3.

The solutions reached by AGOP are also very good. The method is more consistent in finding solutions of the same range for both formulations, although the formulation (2.3) seems to be slightly better handled.

The number of function evaluations needed to solve the problems are quite similar to the ones for CIAO-GO.

2.3.4 LGO

The problem (2.2) was solved using the commercial software LGO, through GAMS as an interface. LGO proposes three modes:

Dataset	Prob. (2.2)	Prob. (2.3)	N.F.(2.2)	N.F.(2.3)	T(2.2)	T(2.3)
aust	19.3285775	19.3765527	1149911	604981	27.969	14.250
breast	0.6253363	0.6231331	455040	318483	7.188	5.078
cleve	1.7616778	1.8009352	498540	473542	4.688	4.344
diab	2.1210358	2.0999435	544576	557624	9.047	9.234
firis	0.1593965	0.1594949	201699	178953	0.484	0.453
iono	0.1355756	0.1222489	1131855	1091260	34.891	33.656
segment	0.9541016	1.0901141	825908	792703	76.078	74.266
vehic	5.8926857	5.9801020	663401	557511	22.031	19.172
wpbc	18.8113554	19.1084435	3415941	1207682	51.734	18.328

Table 3: Results obtained by AGOP

LGO 1 is based on a Branch and Bounds technique

LGO 2 is based on an adaptive random search

LGO 3 is based on a local method with multistart.

Due to bugs in the licensed version of the GAMS system, LGO 1 could not be run on problems whose size was larger than 10 variables. This means that we could only solve the problem for datasets with less than 9 features (In GAMS, the value of the objective function is counted as a variable).

Since the objective function of formulation (2.3) is not defined at the origin, it is not possible to solve this problem using LGO on GAMS. Therefore only the formulation (2.2) was solved.

Only the computational time is indicated in the output of the GAMS software, for this reason the number of function evaluations is not reported.

The results are presented in the following table:

Dataset	LGO 1	LGO 2	LGO 3	T(LGO 1)	T(LGO 2)	T(LGO 3)
aust	-	16.294	18.699	-	3.844	103.390
breast	0.295	0.425	0.554	3.654	4.122	54.045
cleve	-	1.621	1.962	-	1.297	26.937
diab	1.715	1.199	1.585	4.998	5.046	38.287
firis	0.154	0.156	0.159	0.503	0.374	3.568
iono	-	0.106	0.129	-	18.765	352.422
segment	-	0.806	0.866	-	15.343	173.360
vehic	-	5.775	5.861	-	9.829	282.953
wpbc	-	11.956	19.186	-	8.297	249.516

Table 4: Results obtained by LGO on problem (2.2)

It can be concluded that LGO performs relatively well. In two cases it obtained better results than the other methods. In particular, the third method proposed by LGO, which is the simplest one (local method with multistart), seems to perform best, although necessitating much longer computations than the other methods.

2.3.5 Comparison between Relaxation Approach and Global Solvers

Table 5 shows the results obtained for solving the relaxed problem and the original one. It is quite clear that the solution of the relaxed problem is much larger than the solution of the real problem to solve.

Dataset	n_o	n_f	Relaxed	Best	Best software
aust	690	15	3350.01	19.77	CIAO-GO
breast	683	9	1.69996	0.6253	AGOP
cleve	297	13	40.6621	1.962	LGO 3
diab	768	8	78.4567	2.1210358	AGOP
firis	150	4	0.6287	0.1594949	AGOP
iono	351	34	0.1568	0.1355756	AGOP
segment	2310	18	179.6	1.0901141	AGOP
vehic	846	18	302.7	5.9801020	AGOP
wpbc	194	33	1076.2	19.186	LGO 3

Table 5: Summary of the results obtained

3 Second Problem: Minimum of Sum of Squared Distances

3.1 Setting of the Problem

The minimization of the sum of squared distances problem is one of the most widely studied in the literature. This problem finds its roots in data analysis, where it serves as a clustering measurement.

It is locally minimized by the k-means algorithm [5], and many methods have been proposed for solving it.

It can be stated as follows: Given a dataset $A = \{a_i : i = 1, \ldots, n\}$,

$$\begin{array}{c} \text{minimize } \sum_{i=1}^{n} \min_{1 \le j \le q} \|a_i - c_j\|^2 \\ \text{subject to} \\ c_j \in \mathrm{R}^n, 1 \le j \le q \end{array} \tag{3.1}$$

A number of relaxation approaches have been proposed in [10], and successfully applied to two small-sized datasets: the Soybean data (see [7]) and the Bavarian postal codes data (see [12]).

The suite of algorithms GANSO (see [15]) contains several methods for solving general purpose nonconvex nonsmooth optimization problems. Among these methods can be found:

DFBM is a local search based on the discrete-gradient method [2];

D+E stands for DFBM+ECAM. This is a combination of the discrete-gradient method and the Cutting angle method [1]. It is also implemented in the software CIAO-GO [14].

DSO is a software implementation of the AGOP method presented in the previous section

In [10], this problem was studied and solved using a relaxation method. For that purpose, problem (3.1) is rewritten as a 0-1 SDP as follows (see [10] for more details):

Let us define the assignment matrix Y as follows: $y_{ij} = 1$ if a_i is associated to cluster center c_j, and $y_{ij} = 0$ otherwise.

Then, we define:

A is a matrix whose i-th row is a_i;

$Z = X(X^T X)^{-1} X^T$,

I is the identity matrix,

$e = (1, \ldots, 1)^T$.

It is shown in [10] that the problem (3.1) can be rewritten as:

$$\begin{array}{c} \text{minimize } \operatorname{trace}(AA^T(I - Z)) \\ \text{subject to} \\ Ze = e, \operatorname{trace}(Z) = k, Z \geq 0 \end{array} \tag{3.2}$$

$$Z = Z^T, Z^2 = Z, \tag{3.3}$$

where $Z \geq 0$ means that each element of Z is nonnegative.

Then, the proposed relaxation is to replace (3.3) by:

$$z \in \{Z = [z_{ij}] : z_{ij} \leq z_{ii}, z_{ij} + z_{ik} \leq z_{ii} + z_{jk}\},$$

to obtain a linear programming problem.

3.2 Numerical Experiments

The results of the numerical experiments are shown in tables 6 and 7. They are compared with the best known results (column "best known"), found in [10], and with the results of the k-means algorithm. As an indication the best result obtained among the global solvers is given in the column "Best global"

q	dim	Best known	k-means	DFBM	D+E	DSO	Best global
2	70	404.46	404.46	404.46	404.46	404.46	404.46
3	105	215.26	246.46	246.46	246.46	246.46	246.46
4	140	205.96	234.69	234.69	234.69	226	226

Table 6: Results for the Soybean data

4 Conclusions

In this paper we have compared two approaches for solving a nonconvex optimization problem. The first one is to apply direct algorithms. These methods are quite recent, and attempt to directly solve the problem. Although usually able to reach at least a good local minimum, they cannot always guarantee a very good solution for functions having a complex structure, and it is difficult to verify the result.

The second approach is the relaxation: the problem is reformulated, and some constraints are omitted to reach a simpler problem (usually convex or even SDP). Then this relaxed

q	dim	Best known	k-means	DFBM	D+E	DSO	Best global
2	6	$6.03 \cdot 10^{11}$	$7.54 \cdot 10^{11}$	$7.54 \cdot 10^{11}$	$7.54 \cdot 10^{11}$	$6.03 \cdot 10^{11}$	$6.03 \cdot 10^{11}$
3	9	$2.95 \cdot 10^{11}$	$3.54 \cdot 10^{11}$	$3.54 \cdot 10^{11}$	$3.54 \cdot 10^{11}$	$2.95 \cdot 10^{11}$	$2.95 \cdot 10^{11}$
4	12	$1.04 \cdot 10^{11}$	$1.05 \cdot 10^{11}$	$1.05 \cdot 10^{11}$	$1.05 \cdot 10^{11}$	$1.05 \cdot 10^{11}$	$1.05 \cdot 10^{11}$
5	15	$5.98 \cdot 10^{10}$	$7.39 \cdot 10^{10}$	$7.39 \cdot 10^{10}$	$7.39 \cdot 10^{10}$	$5.98 \cdot 10^{10}$	$5.98 \cdot 10^{10}$
6	18	$3.59 \cdot 10^{10}$	$4.60 \cdot 10^{10}$	$4.60 \cdot 10^{10}$	$4.58 \cdot 10^{10}$	$4.58 \cdot 10^{10}$	$3.60 \cdot 10^{10}$
7	21	$2.20 \cdot 10^{10}$	$3.72 \cdot 10^{10}$	$3.72 \cdot 10^{10}$	$3.72 \cdot 10^{10}$	$2.20 \cdot 10^{10}$	$2.20 \cdot 10^{10}$
8	24	$1.34 \cdot 10^{10}$	$3.36 \cdot 10^{10}$	$3.36 \cdot 10^{10}$	$3.24 \cdot 10^{10}$	$1.34 \cdot 10^{10}$	$1.34 \cdot 10^{10}$
9	27	$7.08 \cdot 10^{09}$	$3.17 \cdot 10^{10}$	$3.17 \cdot 10^{10}$	$3.05 \cdot 10^{10}$	$8.42 \cdot 10^{09}$	$8.42 \cdot 10^{09}$

Table 7: Results for the Bavarian postal data

problem is solved and a feasible solution is then constructed using this result. The drawback of these methods is that it is often difficult to reach a good feasible solution from the solution to the relaxed problem.

We have applied and compared both approaches for two different problems arising in the field of data analysis, and carried out numerical experiments over several well-known datasets. The main conclusion from these experiments that no method is universal: each approach proved better than the other ones under different conditions.

4.1 First Experiment

The solution for the relaxed problem is much larger than the best obtained solution for the real problem. As can be seen on table 5, only for one dataset (ionosphere), the results were comparable. For two datasets (Australian Credit and segmentation), the value of the function was more than 150 times larger for the relaxed solution. This may mean that the relaxed solution is in fact quite far from the real solution, and that there may not be any easy way of obtaining one from the other.

The other solvers applied showed a consistent efficiency: they were able to reach a solution close to the global one for many of the problems. Each solver performed better for some of the datasets. However the formulation of the problem proved to be crucial for both CIAO-GO and AGOP: AGOP generally performed better for one formulation than for the other. The fact that this was also dataset-dependent also shows that it is a difficult task to know in advance which formulation will provide better solutions.

This means that the choice of the method or software to solve a problem is very problem-dependent, but also that the formulation of the problem is very method-dependent. In other terms, it is necessary to choose the best formulation in conjunction with the best software.

4.2 Second Experiment

In this case, the relaxation method performs better than the direct ones, at least in the case of small datasets, the reason being that in many cases the result obtained on the convex problem is feasible for the original one. Then this result is a global solution for the problem at hand.

However, these two approaches are of different nature: the complexity of the nonconvex formulation mainly depends on q, the number of clusters: the dimension of the optimization problem is $n \times q$, where n is the number of features.

On the other hand, the dimension of the relaxed problem does not depend on q, but it depends on the number of records in the dataset. As a result, this method is highly

impractical when the dataset is too large.

4.3 Summary

It seems quite difficult to know in advance which formulation suits better which software. A careful study should be undertaken, in order to better understand this aspect.

A parade to this dependency can be applied when the dimension of the problem is not too high: many software allow one to enter a solution as an initial one, or as a benchmark to help the search. A solution can be obtained using one of the methods and entered as an input for another method, in order to be improved.

Nevertheless, it is still beneficial to carry out a prior study of the aspects of the problem, in order to adapt it best to the software (and conversely to chose an appropriate software).

In conclusion, convex relaxation seems mostly useful to provide an lower bound to the global minimum. When the solution obtained appears feasible to the original constraints (as is the case for the minimization of sum-of-squares problem), these method become very efficient, as proved by on the sum-of-square function. However in the general case, general purpose global algorithms provide a good alternative for finding a good feasible solution.

It should also be taken into account that for most optimization problems, the use of a specialized solver can be very beneficial.

Acknowledgments

The author would like to thank Prof. Alex Rubinov and Dr. Musa Mammadov as well as the two anonymous referees for their useful comments and suggestions.

References

[1] M.Y. Andramonov, A.M. Rubinov, and B.M. Glover, Cutting angle for minimizing increasing convex-along-rays functions, Research Report 7/97, SITMS, University of Ballarat, Australia, 1997.

[2] A.M. Bagirov, Numerical methods for minimizing quasidifferentiable functions: a survey and comparison, in *Quasidifferentiability and Related Topics*, V. Demyanov and A. Rubinov (eds.), Kluwer Acad. Publ., Dordrecht, 2000, pp. 33–71.

[3] GAMS Software GmbH, General algebraic modeling system (gams).

[4] M.A. Mammadov, A.M. Rubinov and J. Yearwood, Dynamical systems described by relational elasticities with applications to global optimization, in *Continuous Optimization: Current Trends and Applications*, V. Jeyakumar and A. Rubinov (eds.) Springer, New York, 2005, pp. 365–387.

[5] J. MacQueen, Some methods for classification and analysis of multivariate observations, in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, 1965, pp. 281–297.

[6] Donald Michie, D.J. Spiegelhalter, and C.C. Taylor, *Machine Learning, Neural and Statistical Classification*, Series in Artificial Intelligence, Ellis Horwood, London, 1994.

[7] P.M. Murphy and D.W. Aha, UCI repository of machine learning databases, Technical report, Department of Information and Computer Science, University of California, Irvine, 1992(`http://www.ics.uci.edu/mlearn/MLRepository.html`).

[8] A. Nemirovski, Private communication.

[9] A. Neumaier, O. Shcherbina, W. Huyer, and T. Vinkó, A comparison of complete global optimization solvers, *Math. Program.* 103 (2005) 335–356.

[10] J. Peng and Y. Xia, A new theoretical framework for K-means-type clustering, in *Foundations and Advances in Data Mining*, Chu, W. Chu (edi.) et al., Studies in Fuzziness and Soft Computing 180, Springer, Berlin, 2005, pp. 79–98.

[11] Pintér consulting services, LGO software development.

[12] H. Späth, *Algorithms for Data Reduction and Classification of Objects*, John Wiley and Sons, Ellis Horwood Ltd, 1980.

[13] J.F Sturm, SEDUMI, `http://sedumi.mcmaster.ca/`.

[14] University of Ballarat and Pintér consulting, `http://www.ciao-go.com.au/`.

[15] University of Ballarat, Global Algorithms for NonSmooth Optimization, `http://www.ganso.com.au/`.

Manuscript received 22 December 2005
revised 20 May 2006
accepted for publication 5 June 2006

Julien Ugon
CIAO, University of Ballarat, P.O. Box 661, Ballarat, Vic. 3350, Australia
E-mail address: `j.ugon@ballarat.edu.au`