Linearization Strategy for Boolean Least Squares Problem

M. Safari Moghaddam, M. Salahi
Department of Applied Mathematics, Faculty of Mathematical Sciences
University of Guilan, Rasht, Iran,
salahim@guilan.ac.ir

Abstract: In this paper, we use the linearization strategy using bilinear functions for solving NP-Hard boolean least squares problems. The new model is tighter than the classical linearization strategy. The quality of the solutions obtained by the new linearization strategy is compared with a known algorithm and the bound obtained by semdefinite relaxation showing the better performance of the new approach both in time and quality of the solutions.

Keywords: Boolean least squares, Linearization, Semidefinite relaxation.
AMS Classification: 90C09, 90C22

1. Introduction

Integer least squares (ILS) problem arises in various applications such as position estimation by the Global Positioning System (GPS) [12], maximum likelihood detection of boolean [13], mixed integer version of the least squares problem appears in data fitting applications [17], and may others that are mentioned in [14]. It has been the focus of several recent research. In this paper, we study the boolean version of ILS and call it BLS as follows:

$$\begin{aligned} \min \quad & \|Ax-b\|^2 \\ s.\,t. \quad & x\in\{0,1\}^n, \end{aligned} \tag{1}$$

or

$$\begin{aligned} \min \quad & x^T A^T Ax - 2b^T Ax + b^T b \\ s.\,t. \quad & x\in\{0,1\}^n. \end{aligned}$$

By ignoring the constant term and letting $Q = A^T A$, $c = -2A^T b$, we get the following problem:

$$\begin{aligned} \min \quad & x^T Qx + c^T x \\ s.\,t. \quad & x\in\{0,1\}^n. \end{aligned} \tag{2}$$

It is worth to note that every ILS by introducing new variables can be transformed to BLS (2). In general ILS and BLS are NP-Hard and several approximation algorithms are developed to give upper and lower bounds for its optimal objective value. For example, in [8,9] the authors have proposed a reduction and search algorithm to solve box-constrained ILS and mixed integer least squares. The reduction is based on QR factorization of A. In [15] the author has studied the sphere decoding method in communications and proposed a deterministic method for finding the radius of search sphere. Another widely used approach to deal with hard discrete optimization problems, is the so called semidefinite optimization (SDO) relaxation [2,3]. For example, the known maximum cut problem has been tackled by the SDO relaxation followed by a randomization algorithm led to the 0.87 approximation algorithm [10]. In a most recent work, Park and Boyd used the SDO relaxation to give lower and upper bounds for BLS [14]. One may see several other algorithms for mixed integer quadratic programming in [3, 5-7].

In this paper, we apply the other widely used linearization strategy for solving nonlinear mixed integer programming problems. First, in Section 2 we present the classical linearization strategy. Then in Section 3, we give the linearization using bilinear function [16]. In Section 4, we compare the new linearization strategy with the algorithm of [8] on several randomly generated test problems. The lower bound obtained by SDO relaxation is also provided for all test problems. Finally, some conclusions are given in Section 5.

2. Classical linearization strategy

In this section, we discuss the classical linearization strategy for (2). In order to linearize x^TQx, the following change of variables are done:

$w_{ij} = x_i x_j, \qquad \forall i < j.$

Thus the quadratic term in (2) becomes as follows:

$$x^TQx = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix} \begin{pmatrix} Q_{11} & \cdots & Q_{1n} \\ \vdots & \ddots & \vdots \\ Q_{n1} & \cdots & Q_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

$$= x_1Q_{11}x_1 + x_2Q_{21}x_1 \ldots + x_nQ_{n1}x_1$$
$$+ x_1Q_{12}x_2 + x_2Q_{22}x_2 + \ldots + x_nQ_{n2}x_2$$
$$+ \ldots$$
$$+ x_1Q_{1n}x_n + x_2Q_{2n}x_n + x_nQ_{nn}x_n.$$

Now since $w_{ij} = x_i x_j$, then we have

$$
\begin{aligned}
x^T Qx = {} & x_1 Q_{11} x_1 + x_2 Q_{22} x_2 + \ldots + x_n Q_{nn} x_n \\
& + Q_{21} w_{21} + Q_{31} w_{31} + \ldots + Q_{n1} w_{n1} \\
& + Q_{12} w_{12} + Q_{32} w_{32} + \ldots + Q_{n2} w_{n2} \\
& + \cdots \\
& + Q_{1n} w_{1n} + Q_{2n} w_{2n} + \ldots + Q_{(n-1)n} w_{(n-1)n}.
\end{aligned}
$$

Moreover, since $x_i = 0 \text{ or } 1$, and Q is symmetric, then $x^T Qx = \sum_{i=1}^{n} Q_{ii} x_i + 2\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Q_{ij} w_{ij}$.

Therefore, (2) becomes

$$
\begin{aligned}
\min \ & \left\{ c^T x + \sum_{i=1}^{n} Q_{ii} x_i + 2\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Q_{ij} w_{ij} \right\} \\
s.t. \ & w_{ij} = x_i x_j, \qquad \forall i < j, \\
& x \in \{0,1\}^n.
\end{aligned}
$$

Now we linearize $w_{ij} = x_i x_j$. Since $x_i = 0 \text{ or } 1$, then it is equivalent to

$$w_{ij} = \max\{x_i + x_j - 1, 0\}, \ w_{ij} \in \{0,1\}, \tag{3}$$

and/or

$$w_{ij} = \min\{x_i, x_j\}, \ w_{ij} \in \{0,1\}. \tag{4}$$

Moreover (3) and (4) are both equivalent to the following:

$$w_{ij} \geq x_i + x_j - 1, \qquad w_{ij} \in \{0,1\}, \tag{5}$$

$$w_{ij} \leq x_i, \ w_{ij} \leq x_j, \ \ w_{ij} \in \{0,1\}. \tag{6}$$

Using these, the linearized version of (2) is as follows:

$$
\begin{aligned}
(QP): \ \min \ & \left\{ c^T x + \sum_{i=1}^{n} Q_{ii} x_i + 2\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Q_{ij} w_{ij} \right\} \\
s.t. \ & w_{ij} \geq x_i + x_j - 1, \quad \forall i < j, \\
& w_{ij} \leq x_i, \ \ w_{ij} \leq x_j, \quad \forall i < j, \\
& x, w \in \{0,1\}^n.
\end{aligned} \tag{7}
$$

3. Linearization using bilinear functions

In this section, using bilinear functions we give better linearization compared to (7). First the following problems are solved:

$$\gamma^i_{(\min/\max)} = \min/\max\{Q_i x : x \in \bar{X}\} \qquad \forall i, \tag{8}$$

where Q_i is i-th row of Q and $\bar{X}$ is an appropriate relaxation of X. We further can write (8) as follows:

$$Q=\begin{pmatrix} Q_1 \\ \vdots \\ Q_n \end{pmatrix}, \quad Q_i=\left(q_{i1} \quad \dots \quad q_{in}\right), \quad Q_i x=q_{i1}x_1+\dots+q_{in}x_n,$$
$$\gamma^i_{min/max}=min/max\{q_{i1}x_1+\dots+q_{in}x_n \ : x\in\bar{X}\}.$$

We also let

$$\gamma_{min/max}=\begin{pmatrix} \gamma^1_{min/max} \\ \vdots \\ \gamma^n_{min/max} \end{pmatrix}, \qquad \Gamma_{min/max}=\begin{pmatrix} \gamma^1_{min/max} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \gamma^n_{min/max} \end{pmatrix}.$$

Using these notations and letting $Qx=\gamma$, we can write (2) as follows

$$\begin{aligned} \min \quad & c^T x+x^T\gamma \\ \text{s.}t. \quad & Qx=\gamma, \\ & x\in\{0,1\}^n. \end{aligned} \tag{9}$$

From (8) we have

$$\gamma_{min}\le\gamma\le\gamma_{max}. \tag{10}$$

We further let

$$x_i\gamma_i=s'_i, \qquad \forall i=1,\dots,n. \tag{11}$$

Now multiplying (10) by x_i and $(1-x_i)$, we linearize $x^T\gamma$ as follows:

$$\gamma^i_{\min}x_i\le\gamma_i x_i\le\gamma^i_{\max}x_i,$$
$$\xrightarrow{(11)} \gamma^i_{\min}x_i\le s'_i\le\gamma^i_{\max}x_i, \quad \forall i=1,\dots,n.$$
$$\gamma^i_{\min}(1-x_i)\le\gamma_i(1-x_i)\le\gamma^i_{\max}(1-x_i),$$
$$\xrightarrow{(11)} \gamma^i_{\min}(1-x_i)\le(\gamma_i-s'_i)\le\gamma^i_{\max}(1-x_i), \quad \forall i=1,\dots,n.$$

Therefore (9) becomes

$$\begin{aligned} \min \quad & c^T x+e^T s' \\ s.t. \quad & Qx=\gamma, \\ & \gamma^i_{\min}x_i\le s'_i\le\gamma^i_{\max}x_i, \qquad \forall i, \\ & \gamma^i_{\min}\left(1-x_i\right)\le\left(\gamma_i-s'_i\right)\le\gamma^i_{\max}(1-x_i), \qquad \forall i, \\ & x\in\{0,1\}^n. \end{aligned} \tag{12}$$

We further consider the following change of variables:

$$\begin{aligned} & s_i=s'_i-\gamma^i_{\min}x_i, && \forall i, \\ & y_i=\gamma_i-s'_i-\gamma^i_{\min}(1-x_i), && \forall i. \end{aligned} \tag{13}$$

Now using (13), (12) becomes

$$
\begin{aligned}
(BP):\ \min\quad & c^T x + e^T s + \gamma_{\min}^T x \\
s.t.\quad & Qx = y + s + \Gamma_{\min}\,\mathrm{e}, \\
& 0 \le s_i \le (\gamma_{\max}^i - \gamma_{\min}^i) x_i, \qquad \forall i, \\
& 0 \le y_i \le (\gamma_{\max}^i - \gamma_{\min}^i)(1 - x_i), \quad \forall i, \\
& x \in \{0,1\}^n.
\end{aligned} \tag{14}
$$

As we can easily see, (7) has $O(n^2)$ constraints and variables while (14) has $O(n)$ constraints and variables. This is significantly important when we are dealing with large scale problems.

Theorem 1. Problems (2) and (14) are equivalent in the sense that for each feasible solution to one of them, there exists a feasible solution to the other one which have the same objective value.

Proof: Let $(\bar{x}, \bar{y}, \bar{s})$ be a feasible solution of (14), then as (14) is the result of change of variables in (2), thus $\bar{x}$ is also feasible for (2). By letting $\bar{s}_i{}' = \bar{s}_i + \gamma_{\min}^T \bar{x}$, we can see that they have equal objective values.

4. Numerical experiments

In this section, we compare the efficiency of the linearization strategy with Obils algorithm from [8]. We also provide the lower bounds using the SDO relaxation. Linearized problem is solved using GAMS, Obils is implemented MATLAB and SDO relaxation also is solved using cvx on an Intel G3240, 3.1 GHz machine with 4GB of memory. For each dimension, we have generated 5 instances and the average of running times in seconds and objective values are reported in all tables. In tables, 'Limit' means that the problem is not solved within 3600 seconds.

Example 1: The aim of this example is to compare the classical linearization strategy with the new one. The data for this example are generated as follow and results are summarized in Table 1.

$A = floor\left(100 * rand\left(m,n\right) - 5\right);$

$b = floor\left(100 * rand\left(m,1\right) - 5\right);$

As we see, when the dimension increases, the time required by the classical method increases significantly compared to the two new linearization scheme. Thus for the rest of the tables, we do not include the results of the classical approach.

Table 1. Average computational time of two linearization approaches

m	n	BP	QP
		Average CPU times	
250	200	7.40	245.66
300	250	16.27	902.30
350	300	48.67	Limit

Example 2: The data for this example are generated as follow. The matrix A is generated with density equal to 10% from $[-5 \;\; 5]$ and the vector b also is generated with density equal to 10% from the set $[1 \;\; 100]$. The corresponding Matlab's command are as follow:

$A = floor\left(10 * sprand\left(m, n, 0.1\right) - 5\right);$

$b = floor\left(100 * rand(m,1)\right);$

Table 2. Comparison of average computational times and objective values

m	n	BP		Obils		SDO	
		Average CPU times	Average objective values	Average CPU times	Average objective values	Average CPU times	Average objective values
60	40	0.37	208052.00	4.56	208052.00	1.62	208042.20
80	60	0.49	260560.00	454.05	260560.00	1.95	260551.29
100	80	0.70	310931.40	Limit	N/A	2.01	310929.18

As we see in Table 2, both BP and Obils give the same objective values with significantly different running time. The BP is significantly faster, moreover, for one problem Obils is not able to solve the problem within the time limit . Moreover, the SDO relaxation gives reasonably good lower bounds much faster than Obils.

Example 3: The data for this example are generated as follow and results are summarized in Table 3.

$A = floor\left(100 * rand\left(m, n\right) - 5\right);$

$b = floor\left(100 * rand\left(m, 1\right) - 5\right);$

Table 3. Comparison of average computational times and objective values

m	n	BP		Obils		SDO	
		Average CPU times	Average objective values	Average CPU times	Average objective values	Average CPU times	Average objective values
200	150	7.30	430463.65	3.00	430463.65	4.89	370766.92
250	200	21.35	604489.14	129.29	604489.14	9.51	519603.34
300	250	50.59	685338.38	Limit	N/A	18.15	572943.66

From Table 3, we observe that for the first problem Obils is faster but for the second one it is much slower and it can not solve the last problem in the time limit. Unlike the previous table, the bounds provided by SDO relaxation are much smaller than the optimal values obtained by the linearization approach and Obils.

Example 4: The data for this example are generated as follow and results are summarized in Table 4.

$A = rand(m,n);$

$xc = floor(100 * randn(n,1));$

$b = A * xc + normrnd(0, 0.05, m, 1);$

Table 4. Comparison of average of computational times and objective values

m	n	BP		Obils		SDO	
		Average CPU times	Average objective values	Average CPU times	Average objective values	Average CPU times	Average objective values
100	50	0.47	22705011.96	Limit	N/A	2.33	22675609.40
150	100	1.12	47082389.25	Limit	N/A	5.23	47081244.22
200	150	2.26	68936329.62	Limit	N/A	18.57	68930663.13

Example 5: The data for this example are generated as follow and results are summarized in Table 5.

$A = floor\left(100 * rand\left(m,n\right) - 5\right);$

$b = floor\left(100 * rand\left(m,1\right) - 5\right);$

Table 5. Comparison of average computational times and objective values

m	n	BP		Obils		SDO	
		Average CPU times	Average objective values	Average CPU times	Average objective values	Average CPU times	Average objective values
500	450	203.37	1027785.43	Limit	N/A	158.19	1019546.25
600	550	404.94	2679539.97	Limit	N/A	912.66	2593744.31
700	650	748.16	4106597.93	Limit	N/A	Limit	N/A

5. Conclusions

In this paper, the linearization strategy using bilinear functions is applied to solve the NP-Hard boolean least squares problem. Our experiments on several randomly generated test problems show that this approach performs better than the Obils algorithm and the classical linearization strategy.

References

1. Agrell, E., Eriksson,T., Vardy, A., and Zeger, K., (2002) Closest point search in lattices, IEEE Transaction on Information Theory, vol. 48, pp. 2201–2214.
2. Ben-Tal, A., and Nemirovski, A., (2001) Lectures on Modern Convex Optimization, Theory, Algorithms and Applications. MPS-SIAM series on Optimization, Philadelphia.
3. Billionnet, A., Elloumi, S., and Lambert, A., (2012) Extending the QCR method to general mixed-integer programs. Mathematical Programming, vol. 131, nr.1-2, pp. 381–401.
4. Boyd, S., and Vandenberghe, L., (2004) Convex Optimization. Cambridge University Press.
5. Buchheim, C., and Wiegele., A., (2013) Semidefinite relaxations for non-convex quadratic mixed-integer programming. Mathematical Programming, vol. 141, nr.1-2, pp. 435–452.
6. Buchheim, C., Caprara, A., and Lodi, A., (2012) An effective branch-and-bound algorithm for convex quadratic integer programming. Mathematical Programming, vol. 135, nr. 1-2, pp. 369–395.
7. Buchheim, C., De Santis, M., Palagi, L., and Piacentini, M., (2013) An exact algorithm for nonconvex quadratic integer minimization using ellipsoidal relaxations. SIAM Journal on Optimization, vol. 23, nr. 3, pp. 1867–1889.

8. Chang, X.W., and Han, Q., (2008) Solving box-constrained integer least squares problems. IEEE Transactions on Wireless Communications, vol. 7, nr. 1, pp. 277-287.
9. Chang, X.W., and Zhou, T., (2007) MILES: Matlab package for solving mixed integer least squares problems. GPS Solutions, vol. 11, nr. 4, pp. 289–294.
10. Goemans, M.X., Williamson, D.P., (1995) Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM, vol. 42, nr. 6, pp. 1115-1145.
11. Grant, M., and Boyd, S., (2014) CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx.
12. Hassibi, A., and Boyd, S., (1998) Integer parameter estimation in linear models with applications to GPS. IEEE Transactions on Signal Processing, vol. 46, nr. 11, pp. 2938–2952.
13. Jalden, J., and Ottersten, B., (2005) On the complexity of sphere decoding in digital communications. IEEE Transactions on Signal Processing, vol. 53, nr. 4, pp. 1474–1484.
14. Park, J., and Boyd, S., (2015) A semidefinite programming method for integer convex quadratic minimization. arXiv:1504.07672.
15. Qia, S., (2008) Integer least squares: sphere decoding and LLL algorithm, Proceeding of C3S2E Conference, pp. 23-28.
16. Sherali, H.D., and Smith, J.C., (2007) An improved linearization strategy for zero-one quadratic programming problems. Optimization Letters, vo. 1, nr. 1, pp. 33–47.
17. Ustun, B., and Rudin, C., (2015) Super sparse linear integer models for optimized medical scoring systems. arXiv:1502.04269.