Solving Large Scale Binary Quadratic Problems: Spectral Methods vs. Semidefinite Programming

Carl Olsson
calle@maths.lth.se

Anders P. Eriksson
anderspe@maths.lth.se

Fredrik Kahl
fredrik@maths.lth.se

Centre for Mathematical Sciences
Lund University, Sweden

Abstract

In this paper we introduce two new methods for solving binary quadratic problems. While spectral relaxation methods have been the workhorse subroutine for a wide variety of computer vision problems - segmentation, clustering, image restoration to name a few - it has recently been challenged by semidefinite programming (SDP) relaxations. In fact, it can be shown that SDP relaxations produce better lower bounds than spectral relaxations on binary problems with a quadratic objective function. On the other hand, the computational complexity for SDP increases rapidly as the number of decision variables grows making them inapplicable to large scale problems.

Our methods combine the merits of both spectral and SDP relaxations - better (lower) bounds than traditional spectral methods and considerably faster execution times than SDP. The first method is based on spectral subgradients and can be applied to large scale SDPs with binary decision variables and the second one is based on the trust region problem. Both algorithms have been applied to several large scale vision problems with good performance. [1]

1. Introduction

Spectral relaxation methods can be applied to a wide variety of problems in computer vision. They have been developed to provide solutions to, e.g., image restoration, motion segmentation, partitioning, figure-ground segmentation, clustering, subgraph matching [8, 13]. In particular, large scale problems that can be formulated with a binary quadratic objective function are handled efficiently with several thousands of decision variables.

More recently, semidefinite programming (SDP) relaxations have also been applied to the same type of computer vision problems, e.g., [6, 15, 12]. It can be shown that such relaxations produce better estimates than spectral methods. However, as the number of variables grows, the execution times of the semidefinite programs increase rapidly. In practice, one is limited to a few hundred decision variables.

Spectral and SDP relaxation methods can be regarded as two points on an axis of increasing relaxation performance. We introduce two alternative methods that lie somewhere in between these two relaxations. Unlike standard SDP solvers that suffer from bad time complexity, they can still handle large scale problems. The two methods are based on a subgradient optimization scheme. We show good performance on a number of experimental problems. Our main contributions are:

- An efficient algorithm for solving binary SDP problems with quadratic objective function based on subgradient optimization is developed. In addition, we show how to incorporate linear constraints in the same program.
- The trust region subproblem is introduced and we modify it to in order to be applicable to binary quadratic problems with a linear term in the objective function.

Many of the application problems mentioned above are known to be NP-hard, so in practice they cannot be solved optimally. Thus one is forced to rely on approximate methods which results in sub-optimal solutions. Certain energy (or objective) functionals may be solved in polynomial time, for example, submodular functionals using graph cuts [7], but this is not the topic of the present paper.

In [5], an alternative (and independent) method is derived which is also based on subgradients, called the *spectral bundle method*. Our subgradient method differs from [5] in that it is simpler (just look for an ascent direction) and we have found empirically on the experimental problems (see Section 5) that our method performs equally well (or better). An in-depth comparison of the two alternatives is, however, beyond the scope of this paper.

[1] This work has been funded by the EC's 6th FP SMErobot^{TM}(no. 011838), the Swedish Research Council (no. 2004-4579), SSF programme VISCOS II, and by the Swedish Road Administration and by Vinnova in the project 'Automatisk Bildbehandling och Trafikstudier'.

2. Background

In this paper we study different ways to find approximate solutions of the following binary quadratic problem:

$$z = \inf y^T A y + b^T y, \quad y \in \{-1, 1\}^n \tag{1}$$

where A is an $n \times n$ (possibly indefinite) matrix. A common approach for approximating this highly nonconvex problem is to solve the relaxed problem:

$$z_{sp} = \inf_{||x||^2 = n+1} x^T L x \tag{2}$$

where

$$x = \begin{pmatrix} y \\ y_{n+1} \end{pmatrix}, L = \begin{pmatrix} A & \frac{1}{2}b \\ \frac{1}{2}b^T & 0 \end{pmatrix}.$$

Solving (2) amounts to finding the eigenvector corresponding to the algebraically smallest eigenvalue of L. Therefore we will refer to this problem as the spectral relaxation of (1). The benefits of using this formulation is that eigenvalue problems of this type are well studied and there exist solvers that are able to efficiently exploit sparsity, resulting in fast execution times. A significant weakness of this formulation is that the constraints $y \in \{-1, 1\}^n$ and $y_{n+1} = 1$ are relaxed to $||x||^2 = n + 1$, which usually results in poor approximations.

Now let us turn our attention to bounds obtained through semidefinite programming. Using Lagrange multipliers $\sigma = [\,\sigma_1, \ldots, \sigma_{n+1}\,]^T$ for each binary constraint $x_i^2 - 1 = 0$, one obtains the following relaxation of (1)

$$\sup_{\sigma} \inf_{x} x^T (L + \ \mathrm{diag}(\sigma)) x - e^T \sigma. \tag{3}$$

Here e is an $(n + 1)$-vector of ones. The inner minimization is finite valued if and only if $(L + \mathrm{diag}(\sigma))$ is positive semidefinite, that is, $L + \mathrm{diag}(\sigma) \succeq 0$. This gives the following equivalent relaxation:

$$z_d = \inf_{\sigma} e^T \sigma, \quad L + \mathrm{diag}(\sigma) \succeq 0. \tag{4}$$

We will denote this problem the dual semidefinite problem since it is dual to the following problem (see [3, 6]):

$$z_p = \inf_{X \succeq 0} \mathrm{tr}(LX), \quad \mathrm{diag}(X) = I, \tag{5}$$

where X denotes a $(n+1) \times (n+1)$ matrix. Consequently we will call this problem the primal semidefinite program. Since the dual problems (4) and (5) are convex, there is in general no duality gap. In [6], the proposed method is to solve (5) and use randomized hyperplanes (see [4]) to determine an approximate solution to (1). This method has a number of advantages. Most significantly, using a result from [4] one can derive bounds on the expected value of the relaxed solution. It is demonstrated that the approach works well on a number of computer vision problems. On the other hand, solving this relaxation is computationally expensive. Note that the number of variables is $O(n^2)$ for the primal problem (5) while the original problem (1) only has n variables.

3. A Spectral Subgradient Method

In this section we present a new method for solving the binary quadratic problem (1). Instead of using semidefinite programming we propose to solve the (relaxed) problem

$$z_{sg} = \sup_{\sigma} \inf_{||x||^2 = n+1} x^T (L + \mathrm{diag}(\sigma)) x - e^T \sigma, \tag{6}$$

with steepest ascent. At a first glance it looks as though the optimum value of this problem is higher than that of (3) since we have restricted the set of feasible x. However it is shown in [9] that (3), (5) and (6) are in fact all equivalent. The reason for adding the norm condition to (6) is that for a fixed σ we can solve the inner minimization by finding the smallest eigenvalue. Let

$$\mathcal{L}(x, \sigma) = x^T (L + \mathrm{diag}(\sigma)) x - e^T \sigma \tag{7}$$

$$f(\sigma) = \inf_{||x||^2 = n+1} \mathcal{L}(x, \sigma). \tag{8}$$

Since f is a pointwise infimum of functions linear in σ it is easy to see that f is a concave function. Hence our problem is a concave maximization problem. Equivalently, f can be written as

$$f(\sigma) = (n + 1)\lambda_{min}(L + \mathrm{diag}(\sigma)) - e^T \sigma. \tag{9}$$

Here λ_{min} denotes the smallest eigenvalue. It is widely known that the eigenvalues are analytic (and thereby differentiable) functions everywhere except where they cross. In order to be able to use a steepest ascent method we need to consider subgradients as eigenvalues *will* cross during the optimization. Recall the definition of a subgradient [1].

Definition 1. *If $f : \mathbb{R}^{n+1} \mapsto \mathbb{R}$ is concave, then $\xi \in \mathbb{R}^{n+1}$ is a subgradient to f at σ_0 if*

$$f(\sigma) \leq f(\sigma_0) + \xi^T (\sigma - \sigma_0), \quad \forall \sigma \in \mathbb{R}^{n+1}. \tag{10}$$

One can show that if a function is differentiable then the gradient is the only vector satisfying (10). We will denote the set of all subgradients at a point σ_0 by $\partial f(\sigma_0)$. From (10) it is easy to see that this set is convex and if $0 \in \partial f(\sigma_0)$ then σ_0 is a global maximum. Next we show how to calculate the subgradients of our problem. Let $\bar{x}^2$ be the vector containing the entries of $\bar{x}$ squared. Then we have:

Lemma 1. *If $\bar{x}$ is an eigenvector corresponding to the minimal eigenvalue of $L + diag(\bar{\sigma})$ with norm $||\bar{x}||^2 = n + 1$ then $\xi = \bar{x}^2 - e$ is a subgradient of f at $\bar{\sigma}$.*

Proof. If $\bar{x}$ is an eigenvector corresponding to the minimal eigenvalue of $L + \text{diag}(\bar{\sigma})$ then $\bar{x}$ solves

$$\inf_{||x||^2=n+1} \mathcal{L}(x, \bar{\sigma}). \tag{11}$$

Assume that $\tilde{x}$ solves

$$\inf_{||x||^2=n+1} \mathcal{L}(x, \tilde{\sigma}) \tag{12}$$

then

$$\begin{aligned} f(\tilde{\sigma}) &= \tilde{x}^T(L + \text{diag}(\tilde{\sigma}))\tilde{x} - e^T\tilde{\sigma} \\ &\leq \bar{x}^T(L + \text{diag}(\tilde{\sigma}))\bar{x} - e^T\tilde{\sigma} \\ &= f(\bar{\sigma}) + \bar{x}^T\text{diag}(\tilde{\sigma} - \bar{\sigma})\bar{x} - e^T(\tilde{\sigma} - \bar{\sigma}) \\ &= f(\bar{\sigma}) + \sum_i (\tilde{\sigma}_i - \bar{\sigma}_i)(\bar{x}_i^2 - 1) \\ &= f(\bar{\sigma}) + \xi^T(\tilde{\sigma} - \bar{\sigma}). \end{aligned}$$

The inequality comes from the fact that $\tilde{x}$ solves (12). □

The result above is actually a special case of a more general result given in [1] (Theorem 6.3.4). Next we state three corollaries obtained from [1] (Theorems 6.3.7, 6.3.6 and 6.3.11). The first one gives a characterization of all subgradients.

Corollary 1. *Let $\mathcal{E}(\bar{\sigma})$ be the set of all eigenvectors with norm $(n+1)^2$ corresponding to the minimal eigenvalue of $L + diag(\bar{\sigma})$. Then the set of all subgradients of f at $\bar{\sigma}$ is given by*

$$\partial f(\bar{\sigma}) = \textit{convhull}(\{x^2 - e;\ x \in \mathcal{E}(\bar{\sigma})\}). \tag{13}$$

We do not give the proof here but note that the inclusion $\partial f(\bar{\sigma}) \supseteq \text{convhull}(\{x^2 - e;\ x \in \mathcal{E}(\bar{\sigma})\})$ is obvious by Lemma 1 and the fact that $\partial f(\bar{\sigma})$ is a convex set.

Corollary 2. *Let $\mathcal{E}(\bar{\sigma})$ be the set of all eigenvectors with norm $(n+1)^2$ corresponding to the minimal eigenvalue of $L + diag(\bar{\sigma})$. Then*

$$f'(\bar{\sigma}, d) = \inf_{\xi \in \partial f(\bar{\sigma})} d^T\xi = \inf_{x \in \mathcal{E}(\bar{\sigma})} d^T(x^2 - e). \tag{14}$$

Here $f'(\bar{\sigma}, d)$ is the directional derivative in the direction d or formally

$$f'(\bar{\sigma}, d) = \lim_{t \to 0^+} \frac{f(\bar{\sigma} + td) - f(\bar{\sigma})}{t}. \tag{15}$$

The first equality is proven in [1]. The second equality follows from Corollary 1 and the fact that the objective function $d^T\xi$ is linear in ξ. For a linear (concave) function the optimum is always attained in an extreme point. From [1] we also obtain the following

Corollary 3. *The direction d of steepest ascent at σ_0 is given by*

$$d = \begin{cases} 0 & if\ \ \xi = 0 \\ \frac{\xi}{||\xi||} & if\ \ \xi \neq 0 \end{cases} \tag{16}$$

where $\xi \in \partial f(\sigma_0)$ is the subgradient with smallest norm.

3.1. Implementation

The basic idea is to find an ascending direction and then to solve an approximation of $f(\sigma)$ along this direction.

3.1.1 Finding ascent directions

The first step is to find an ascending direction. We use Corollary 1 to find a good direction. A vector $x \in \mathcal{E}(\bar{\sigma})$ can be written

$$x = \sum_i \lambda_i x_i, \quad \sum_i \lambda_i^2 = 1, \tag{17}$$

where $\{x_i\}$ is an orthogonal base of the eigenspace corresponding to the smallest eigenvalue (with $||x_i||^2 = n+1$). For the full subgradient set we need to calculate $x^2 - e$ for all possible values of λ in (17). In practice, we are led to an approximation and empirically we have found that it is enough to pick the vectors $x_i^2 - e$ and use the convex envelope of these vectors as our approximation. Let $\mathcal{S}$ be our approximating set. To determine the best direction, the vector of minimum norm in $\mathcal{S}$ needs to be found. The search can be written as

$$\inf_{\xi \in \mathcal{S}} ||\xi||^2 = \inf ||\sum_k \mu_k x_k^2 - e||^2, \ \sum_k \mu_k = 1, \mu_k \geq 0, \tag{18}$$

which is a convex quadratic program in μ_k that can be solved efficiently. To test if an ascending direction d is actually obtained, we use Corollary 2 to calculate the directional derivative. In fact we can solve the optimization problem (14) efficiently by using the parameterization (17), which results in

$$\inf d^T \left((\sum_i \lambda_i x_i)^2 - e \right), \ \sum_i \lambda_i^2 = 1. \tag{19}$$

This is a quadratic function in λ with a norm constraint which can be solved by calculating eigenvalues. If d is not an ascent direction then we add more vectors to the set $\mathcal{S}$ to improve the approximation. In this way we either find an ascending direction or we find that zero is a subgradient, meaning that we have reached the global maximum.

3.1.2 Approximating f along a direction

The next step is to find an approximation $\tilde{f}$ of the objective function along a direction. We do this by restricting the set of feasible x to a set X consisting of a few of the eigenvectors corresponding to the lowest eigenvalues of $L + \text{diag}(\sigma)$. The intuition behind this choice for X is that if the eigenvalue λ_i is distinct then $x_i^2 - e$ is in fact the gradient of the function

$$(n+1)\lambda_i(L + \text{diag}(\sigma)) - e^T\sigma, \tag{20}$$

where $\lambda_i(\cdot)$ is the ith smallest eigenvalue as a function of a matrix. The expression

$$f_i(t) = x_i^T(L + \text{diag}(\bar{\sigma} + td))x_i - e^T(\bar{\sigma} + td) \tag{21}$$

is then a Taylor expansion around $\bar{\sigma}$ in the direction d. The function f_1 approximates f well in neighborhood around $t = 0$ if the smallest eigenvalue do not cross any other eigenvalue. If it does then one can expect that there is some i such that $\min(f_0(\sigma), f_i(\sigma))$ is a good approximation. This gives us a function $\bar{f}$ of the type

$$\bar{f}(\sigma) = \inf_{x_i \in X} x_i^T (L + \text{diag}(\bar{\sigma} + td)) x_i - e^T(\bar{\sigma} + td). \quad (22)$$

To optimize this function we can solve the linear program

$$\begin{aligned} &\max_{t,f} f \\ &f \leq x_i^T (L + \text{diag}(\bar{\sigma} + td)) x_i - e^T(\bar{\sigma} + td) \\ &\forall x_i \in X,\ t < t_{max}. \end{aligned} \quad (23)$$

The parameter t_{max} is used to express the interval for which the approximation is valid. The program gives a value for t and thereby a new $\tilde{\sigma} = \bar{\sigma} + td$. In general, the new σ gives a higher value when evaluating f at $\tilde{\sigma}$. However if the approximation is not good enough, one needs to improve the approximating function. This can be accomplished by making a new Taylor expansion around the point $\tilde{\sigma}$ and add these terms to our approximation and repeat the process. Figure 1 shows two examples of the approximating function.

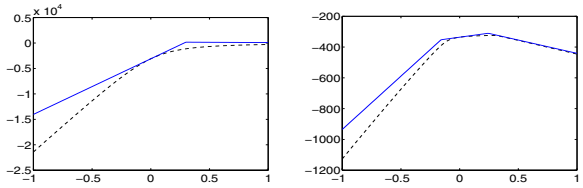

Figure 1. Two approximations of the objective function along ascent directions. The dashed line is the true objective function and the solid line is the approximation.

4. The Trust Region Problem

An interesting relaxation is obtained if we drop the constraints that $y_i \in \{-1, 1\}$ and only require that the last variable satisfies $y_{n+1} = 1$. We then obtain a problem similar to the trust region problem:

$$z_{tr} = \inf_{||y||^2 = n} y^T A y + b^T y. \quad (24)$$

We propose to use this relaxation instead of the spectral relaxation (2). Since the objective function is the same as for the spectral relaxation with $y_{n+1} = 1$ it is obvious that

$$z_{sp} \leq z_{tr} \quad (25)$$

holds. Equality will only occur if the solution to z_{sp} happens to have ± 1 as its last component. This is generally not the case. In fact, empirically we have found that the last component is often farther away from ± 1 than the rest of the components. So by enforcing the constraint, that is, solving (24) often yields much better solutions.

The trust region problem have been studied extensively in the optimization literature. A remarkable fact is that this is a nonconvex problem with no duality gap (see [3]). We could solve it with the same type of subgradient algorithm as developed in the previous section, however, there already exist efficient large scale solvers. It is well-know that y is a global minimizer if there exists $\lambda \in \mathbb{R}$ such that

$$(A - \lambda I)y = -\tfrac{1}{2}b \quad (26)$$

$$||y||^2 = n \quad (27)$$

$$(A - \lambda I) \succeq 0. \quad (28)$$

Here (26), (27) are the KKT conditions for stationarity, while (28) determines the global minimizer (see [10],[11] and references therein). It can be solved using semidefinite programming [10], however, the LSTRS-algorithm developed in [11] is more efficient. An implementation is a publicly available from the authors of [11] upon request. The algorithm solves

$$\inf_{||y||^2 \leq n} y^T A y + b^T y. \quad (29)$$

This is a slightly different problem than our formulation of the trust region problem. However it is easy to see that by adding $k(n - y^T y)$ for sufficiently large k the obtained solution will satisfy $||y||^2 = n$. LSTRS works by solving a parametrised eigenvalue problem. It searches for an α such that the eigenvalue problem

$$\begin{pmatrix} A & b/2 \\ b^T/2 & \alpha \end{pmatrix} \begin{pmatrix} y \\ 1 \end{pmatrix} = \lambda_{min} \begin{pmatrix} y \\ 1 \end{pmatrix} \quad (30)$$

or equivalently

$$\begin{aligned} (A - \lambda_{min} I)y &= -\frac{1}{2}b \\ \alpha - \lambda_{min} &= -\frac{1}{2}b^T y \end{aligned} \quad (31)$$

has a solution. Finding this α is done by determining a λ such that $\phi'(\lambda) = n$, where ϕ is defined by

$$\phi(\lambda) = \frac{1}{4}b^T(A - \lambda I)^\dagger b = -\frac{1}{2}b^T y. \quad (32)$$

It can be shown that λ gives a solution to (31). ϕ is a rational function with poles at the eigenvalues of A and can therefore be expensive to compute. Instead rational interpolation is used to efficiently determine λ. For further details see [11].

5. Experiments

In this section we evaluate the performance of our methods for a few different applications that can be solved as

binary quadratic problems. The algorithms are compared with spectral relaxations using Matlab's sparse eigenvalue solver, SDP relaxations using SeDuMi [14] and the spectral bundle algorithm developed by Helmberg [5]. Our spectral subgradient algorithm is implemented in Matlab and the trust region algorithm is based on LSTRS [11] (also Matlab). Note that our implementations consist of simple matlab scripts while the other software has implementations in c (and often highly optimized for speed).

5.1. Binary Restoration

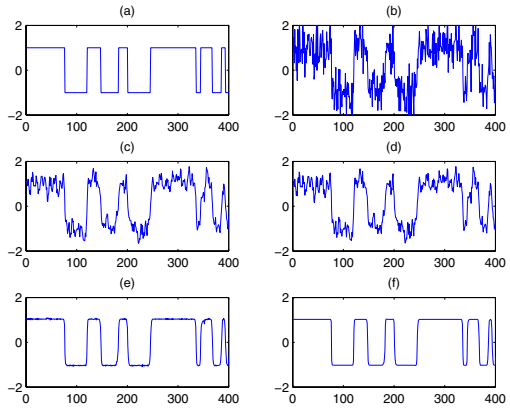

Figure 2. Computed solutions for the signal problem with $n = 400$. (a) Original signal, (b) signal + noise, (c) solution obtained using spectral relaxations, (d) trust region, (e) subgradient algorithm and (f) dual semidefinite program.

As a test problem (which can be solved exactly by other means), we first consider the problem of separating a signal from noise. The signal $\{x_i\}$, $i = 1, ..., n$ is assumed to take the values ± 1. Normally distributed noise with mean 0 and variation 0.6 is then added to obtain a noisy signal $\{s_i\}$, $i = 1, ..., n$. Figure 2 (a) and (b) graphs the original signal and the noisy signal respectively for $n = 400$. A strategy to recover the original signal is to minimize the following objective function:

$$\sum_i (x_i - s_i)^2 + \mu \sum_i \sum_{j \in N(i)} (x_i - x_j)^2, \quad x_i \in \{-1, 1\}. \tag{33}$$

Here $N(i)$ means a neighborhood of i, in this case $\{i - 1, i + 1\}$. By adding the (homogenization) variable x_{n+1}, the problem can be transformed to the same form as in (6). Table 1 shows the execution times and Table 2 displays the obtained estimates for different n. For the subgradient method, 10 iterations were run and in each iteration, the 15 smallest eigenvectors were computed for the approximation set $\mathcal{S}$ in (18). Note in particular the growth rate of the execution times for the SDP. Figure 2 (b) - (d) shows the computed signals for the different methods when $n = 400$. The results for other values of n have similar appearance. The spectral relaxations behave (reasonably) well for this problem as the estimated last x_{n+1} happens to be close to ± 1.

n	*Spectral*	*Trust region*	*Subgradient*	*SDP*
100	0.33	0.60	4.21	3.81
200	0.30	0.62	6.25	13.4
400	0.32	0.68	6.70	180
600	0.33	0.80	10.7	637
800	0.49	1.40	10.1	2365
1000	0.37	1.85	15.2	4830

Table 1. Execution times in seconds for the signal problem.

n	*Spectral*	*Trust region*	*Subgradient*	*SDP*
100	24.3	31.6	40.6	53.1
200	27.4	40.5	53.5	76.1
400	74.9	88.4	139	174
600	134	164	240	309
800	169	207	282	373
1000	178	229	322	439

Table 2. Objective values of the relaxations. A higher value means a better lower bound for the (unknown) optimal value.

Next we consider a similar problem as above, which was also a test problem in [6]. We want to restore the map of Iceland given in Figure 3. The objective function is the same

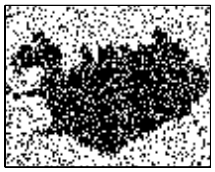

Figure 3. Map of Iceland corrupted by noise.

as in (33), except that the neighborhood of a pixel is defined to be all its four neighboring pixels. The size of the image is 78×104, which yields a program with $78 \cdot 104 + 1 = 8113$ variables. Recall that the semidefinite primal program will contain $8113^2 = 65820769$ variables and therefore we have not been able to compute a solution with SeDuMi. In [6], a different SDP solver was used and the execution time was 64885s. Instead we compare with the spectral bundle algorithm [5]. Table 3 gives the execution times and the ob-

Method	Time (s)	Lower bound
Spectral	0.48	-1920
Trust region	2.69	-1760
Subgradient, 10 iter.	74.6	-453
Bundle, 5 iter.	150.4	-493

Table 3. Execution times and objective values of the computed lower bounds for the Iceland image.

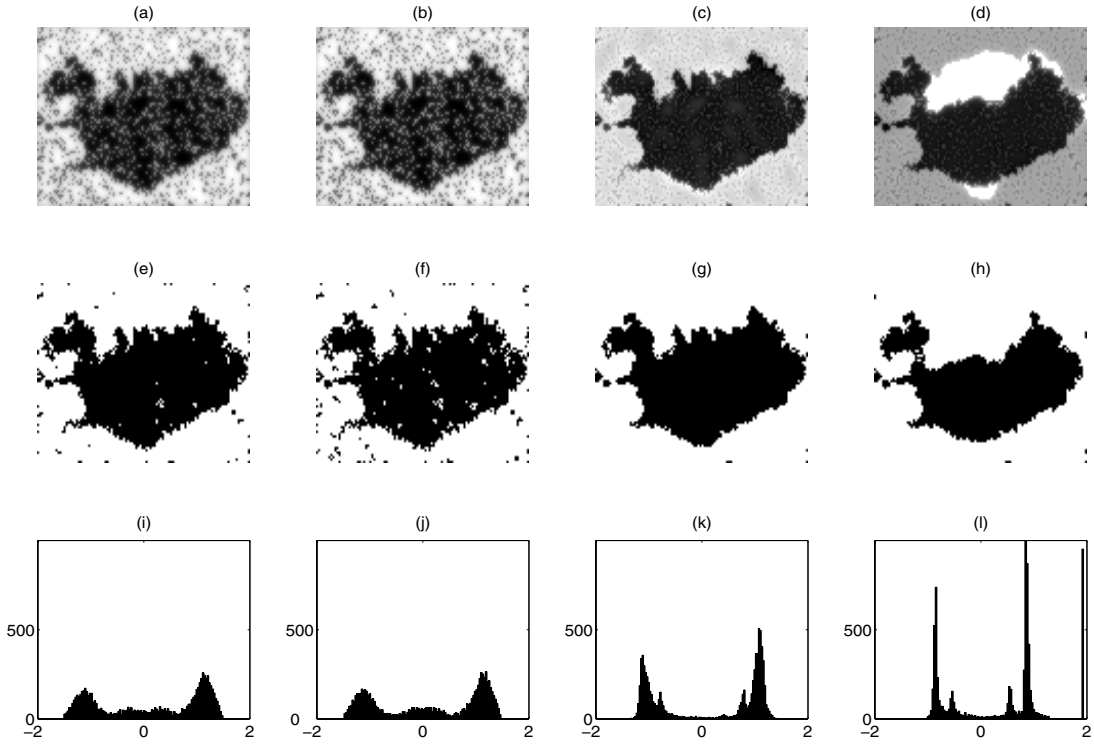

Figure 4. Top row: relaxed soutions. Middle: threshholded solutions. Bottom: histogram of the estimated pixel values. (a),(e),(i): spectral method, (b),(f),(j): trust region, (c),(g),(k): subgradient, 10 iterations, (d),(h),(l): Helmberg's bundle method, 5 iterations.

jective values of the estimations. Figure 4 shows the resulting restorations for the different methods. For the subgradient algorithm, the 4 smallest eigenvalues were used in (18). Even though the spectral relaxation results in a slightly lower objective value than the trust region, the restoration looks just as good. Here the last component of the eigenvector is 0.85 which explains the similarity of these two restorations. The subgradient method yields a solution with values closer to ± 1 as expected. Recall that there is a duality gap which means that the optimal solution will not attain $x_i = \pm 1$ for all i in general. The spectral bundle method provides a solution where some pixel values are much larger than 1. In order to make the difference between pixels with values -1 and 1 visible in Figure 4(d) we hade to replace these pixel values with a lower value. This results in the white areas in Figure 4(d) and the bar close to the value 2 in Figure 4(d).

5.2. Partitioning

In this section we consider the problem of partitioning an image into perceptionally different parts. Figure 5 (a) shows the image that is to be partitioned. Here we want to separate the buildings from the sky. To do this we use the following regularization term

$$\sum_{ij} w_{ij}(x_i - x_j)^2. \tag{34}$$

The weights w_{ij} are of the type

$$w_{ij} = e^{-\frac{(\mathrm{RGB}(i)-\mathrm{RGB}(j))^2}{\sigma_{\mathrm{RGB}}}} e^{\frac{-d(i,j)^2}{\sigma_d}}, \tag{35}$$

where $\mathrm{RGB}(i)$ denotes the RGB value of pixel i and $d(i,j)$ denotes the distance between pixels i and j. To avoid solutions where all pixels are put in the same partition, and to favour balanced partitions, a term penalizing unbalanced solutions is added. If one adds the constraint $e^T x = 0$ (as in [6]) or equivalently $x^T e e^T x = 0$ we will get partitions of exactly equal size (at least for the subgradient method). Instead we add a penalty term to the objective function yielding a problem of the type

$$\inf x^T(L + \mu e e^T)x, \quad x_i \in \{-1, 1\}. \tag{36}$$

Observe that this problem is not submodular [7]. Since the size of the skyline image (Figure 5(a)) is 35×55 we obtain a dense matrix of size 1925×1925. However, because of the structure of the matrix it is easy to calculate $(L + \mu e e^T)x$ which is all that is needed to employ power iteration type procedures to calculate eigensystems. This type of matrices are not supported in the spectral bundle software, so

we cannot compare with this method. Also, the problem is too large for SeDuMi and there is no point in running the trust region method on this problem since the matrix L has not been homogenized. Figure 5 (b) shows the resulting partition. Figures 5 (e),(f) give the relaxed solutions after 4 and 7 iterations, respectively, of the subgradient algorithm. Both relaxed solutions yield the same result when thresholded at zero. As a comparison, we have included the partitionings obtained from Normalized Cuts [13] which is a frequently applied method for segmentation. The reason for the strange partitioning in Figures 5(c),(d) is that the Fiedler vector in Normalized Cuts essentially contains values close to -0.3 and 3.3 and the median is also close to -0.3. Table 4 shows the computing times of the different methods. Note that the convergence of the subgradient method here is slower than previously. This is because the eigenvalue calculations is more demanding for $(L + \mu ee^T)$.

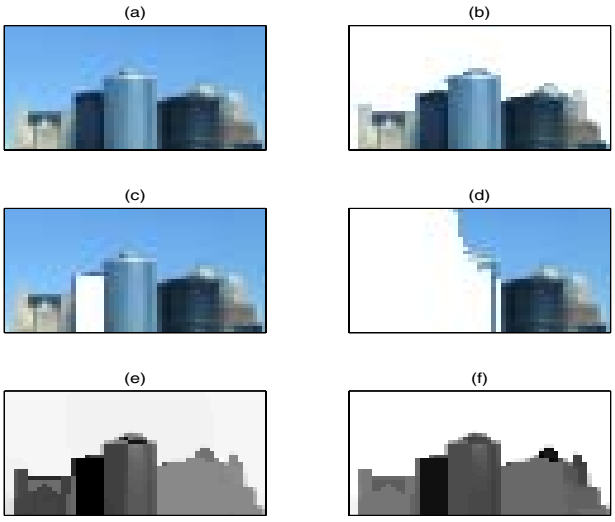

Figure 5. (a) Original image, (b) thresholded segmentation with 7 iterations of the subgradient algorithm (white pixels correspond to one class, remaining pixels are in the other class) (c) Fiedlervector thresholded at the median, (d) Fiedlervector thresholded at the mean, (e),(f) relaxed (untruncated) solutions obtained with 4 and 7 iterations, respectively, of the subgradient algorithm.

Method	Time (s)
Subgradient, 4 iter.	209
Subgradient, 7 iter.	288
Normalized Cuts	5.5

Table 4. Computing times for the skyline image.

6. Registration

In our final experiments we consider the registration problem. It appears as a subproblem in many vision applications and similar formulations as the one we propose here have appeared in [2, 12, 15].

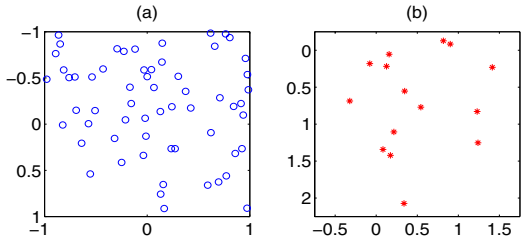

Figure 6. One random example for the registration problem: (a) Target points $n = 60$ and (b) source points $m = 15$.

Suppose we are given a set of m source points that should be registered to a set of n target points, where $m < n$. Let x_{ij} denote a binary $(0, 1)$-variable which is 1 when source point i is matched to target point j, otherwise 0. As objective function, we choose the quadratic function

$$\sum w_{ijkl} x_{ij} x_{kl}, \tag{37}$$

and set $w_{ijkl} = -1$ if the coordinates of the source points s_i, s_k are *consistent* with the coordinates of the target points t_j, t_l, otherwise $w_{ijkl} = 0$. Two correspondence pairs are considered to be consistent if the distances are approximately the same between source and target pairs, that is,

$$\text{abs}(||s_i - s_k|| - ||t_j - t_l||) < \theta, \tag{38}$$

for some threshold θ. Each source point is a priori equally likely to be matched to any of the target points and hence there is no linear term in the objective function. In addition, each source point should be mapped to one of the target points and hence $\sum_j x_{ij} = 1$ for all i. Also, two source points cannot be mapped to the same target point. This can be specified by introducing $(0, 1)$-slack variables $x_{m+1,j}$ for $j = 1, \ldots, n$ and the constraints $\sum_j x_{m+1,j} = n - m$ as well as $\sum_{i=1}^{m+1} x_{ij} = 1$ for all j.

By substituting $x_{ij} = \frac{z_{ij}+1}{2}$, the problem is turned into a standard $(-1, 1)$-problem, but now with linear equality constraints. In the case of the trust region method we may penalize deviations from the linear constraints by adding penalties of the type $\mu(\sum_j x_{ij} - 1)^2$ to the objective function. One could the same in the case of the subgradient algorithm, however, in this case the penalties have to be homogenized and may therefore not be as effective as for the trust region method. Instead Lagrange multipliers of the type $\sigma_k(\sum_j x_{ij})^2 - \sigma_k$ are introduced. These multipliers can then be handled in exactly the same way as the constraints $x_{ij}^2 - 1 = 0$. Each constraint gives a new entry in the subgradient vector which is updated in the same way as before.

We have tested the formulation on random data of various sizes. First, coordinates for the n target points are randomly generated with a uniform distribution, then we randomly selected m source points out of the target points,

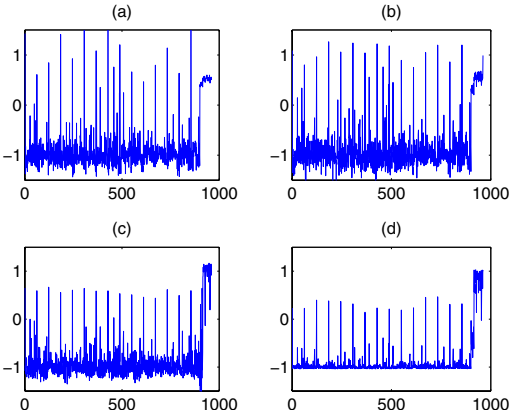

Figure 7. Computed solutions $z = [\, z_{11}, z_{12}, \ldots, z_{m+1,n} \,]$ for the registration problem using (a) the trust region method, (b) the subgradient method, 7 iterations, (c) the subgradient method, 15 iterations, and (d) SDP with SeDuMi, cf. Figure 6.

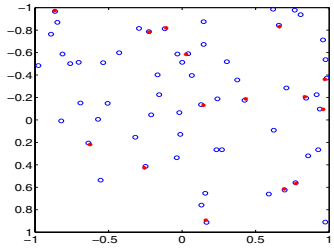

Figure 8. Registration of the source points to their corresponding target points, cf. Figure 6.

Method	Time (s)
Trust region	1.9
Subgradient, 7 iter.	43.5
Subgradient, 15 iter.	193
SDP	6867

Table 5. The registration problem with $m = 15, n = 60$.

added noise and applied a random Euclidean motion. Figures 6 (a),(b) show the target and source points for one example with $m = 15$ and $n = 60$. The threshold θ is set to 0.1. The untruncated (vectorized) solutions for z_{ij} are plotted in Figure 7 and the resulting registration for the subgradient method is shown in Figure 8. The standard spectral relaxation for this problem works rather bad as the last entry z_{n+1} is in general far from one. The computing times are given in Table 5. Note that this example has approximately four times as many decision variables as the largest problems dealt with in [12, 15]. For more information on the quality of SDP relaxations for this problem, the reader is also referred to the same papers.

7. Conclusions

We have shown how large scale binary problems with quadratic objectives can be solved by taking advantage of the spectral properties of such problems. The approximation gap compared to traditional spectral relaxations is considerably smaller, especially, for the subgradient method. Compared to standard SDP relaxations, the computational effort is less demanding, in particular, for the trust region method. Future work includes to apply the two methods to more problems that can be formulated within the same framework and to make an in-depth experimental comparisons. It would also be interesting to see how the proposed methods behave in a branch-and-bound algorithm for obtaining more accurate estimates.

References

[1] S. Bazaraa, Sherali. *Nonlinear Programming, Theory and Algorithms.* Wiley, 2006.

[2] A. Berg, T. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *Conf. Computer Vision and Pattern Recognition*, pages 26–33, San Diego, USA, 2005.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] M. Goemans and D. Wiliamson. Improved approximation algorithms for maximum cut and satisfiability problem using semidefinite programming. *J.ACM*, 42(6):1115–1145, 1995.

[5] C. Helmberg and F. Rendl. A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3):673–696, 2000.

[6] J. Keuchel, C. Schnörr, C. Schellewald, and D. Cremers. Binary partitioning, perceptual grouping, and restoration with semidefinite programming. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(11):1364–1379, 2006.

[7] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.

[8] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, 2002.

[9] S. Poljak, F. Rendl, and H. Wolkowicz. A recipe for semidefinite relaxation for (0,1)-quadratic programming. *Journal of Global Optimization*, 7:51–73, 1995.

[10] F. Rendl and H. Wolkowicz. A semidefinite framework for trust region subproblems with applications to large scale minimization. *Math. Prog.*, 77(2 Ser.B):273–299, 1997.

[11] M. Rojas, S. Santos, and D. Sorensen. A new matrix-free algorithm for the large-scale trust-region subproblem. *SIAM Journal on optimization*, 11(3):611–646, 2000.

[12] C. Schellewald and C. Schnörr. Probabilistic subgraph matching based on convex relaxation. In *EMMCVPR*, pages 171–186, 2005.

[13] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[14] J. Sturm. Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11-12:625–653, 1999.

[15] P. Torr. Solving markov random fields using semi definite programming. In *Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.