Computer Vision and Image Understanding 112 (2008) 3–13

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Improved spectral relaxation methods for binary quadratic optimization problems

Carl Olsson *, Anders P. Eriksson, Fredrik Kahl

Lund University, Centre for Mathematical Sciences, Box 118, 22100 Lund, Sweden

ARTICLE INFO

Article history:
Received 25 October 2007
Accepted 24 May 2008
Available online 8 June 2008

Keywords:
Quadratic binary optimization
Spectral relaxation
Image partitioning
Subgraph matching
Trust region problem
Semidefinite programming
Discrete optimization
Binary restoration

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, subgraph matching 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. Introduction

Spectral methods can be applied to a wide variety of problems in computer vision. They have been developed to provide solutions to, for example, motion segmentation, figure–ground segmentation, clustering, subgraph matching and digital matting [18,16,24,30,14]. 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, for example, [12,29,23]. 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 poor 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 problems. Experimental results are given on the following problems: (i) image segmentation with prior information, (ii) binary restoration, (iii) partitioning and (iv) subgraph matching. Our main contributions are:

- An efficient algorithm for solving binary SDP problems with quadratic objective functions 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 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 result in suboptimal solutions. Other methods for relaxing binary quadratic optimization problems with linear constraints do exist. For instance, [5] proposes a heuristic, applied to Markov random fields, for finding solutions that are better than the ones provided by spectral relaxation. Certain energy (or objective) functionals may be solved in polynomial time, for example, submodular functionals using graph cuts [13]. However, this is not the topic of the present paper. Our primary interest is non-submodular functionals.

In [11], an alternative (and independent) method is derived which is also based on subgradients, called the *spectral bundle method*. Our subgradient method differs from [11] 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.

* Corresponding author.
E-mail address: calle@maths.lth.se (C. Olsson).

1077-3142/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2008.05.010

1.1. Outline

The outline of this paper is as follows. In the next section, we present the problem and some existing approximation techniques for obtaining approximate solutions.

In Section 3, we present our algorithm. We develop theory for improving the standard spectral relaxation by using the notion of subgradients. A subgradient can be regarded as a generalization of a standard gradient which is applicable when a function is not differentiable. We show that for our problem the subgradients can be calculated analytically. This gives way for determining ascent directions of the objective function.

In Section 4, we study the Trust Region Subproblem, which is interesting special case of a binary quadratic relaxation in which we only try to enforce the binary constraints on one of the decision variables. This has been extensively studied in the optimization literature and we show that it is always possible to solve this problem exactly.

Finally, we test our algorithms and compare with existing methods on the following problems: image segmentation with prior information, binary restoration, partitioning and subgraph matching. Preliminary results of this work were presented in [17,7].

2. Background

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

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

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

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

where

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

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 in the matrix L, 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 often results in poor approximations.

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

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

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:

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

We will denote this problem the dual semidefinite problem since it is dual to the following problem (see [3,12]):

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

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 [12], the proposed method is to solve (5) and use randomized hyperplanes (see [10]) to determine an approximate solution to (1). This method has a number of advantages. Most significantly, using a result from [10] 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

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

with steepest ascent. At a first glance it looks as though the optimum value of this problem is greater than that of (3) since we have restricted the set of feasible x. However it is shown in [19] 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.

3.1. Differentiating the objective function

Let

$$\mathscr{L}(x, \sigma) = x^{\mathrm{T}}(L + \mathrm{diag}(\sigma))x - e^{\mathrm{T}}\sigma \tag{7}$$
$$f(\sigma) = \inf_{\|x\|^2 = n+1} \mathscr{L}(x, \sigma). \tag{8}$$

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

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

Here, $\lambda_{\min}(\cdot)$ denotes the smallest eigenvalue of the entering matrix. It is widely known that the eigenvalues are analytic (and thereby differentiable) functions everywhere as long as they are distinct. 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 3.1. 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

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

Fig. 1 shows a geometrical interpretation of (10). Note that if f is differentiable at σ_0, then letting ξ be the gradient of f turns the right hand side of (10) into the tangent plane. One can show that if a function is differentiable then the gradient is the only vector satisfying (10). If f is not differentiable at σ_0 then there are several subgradients 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 at a global maximum.

Next, we show how to calculate the subgradients of our problem. Let x^2 be the vector containing the entries of x squared. Then we have:

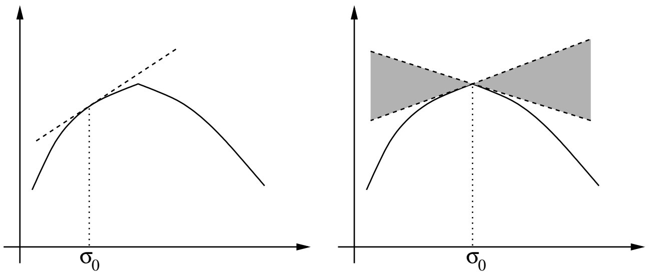


Fig. 1. Geometric interpretation of the definition of subgradients. Left: When the function is differentiable in σ_0 the only possible right hand side in (10) is the tangent plane. Right: When the function is not differentiable there are several planes fulfilling (10), each one giving rise to a subgradient.

Lemma 3.2. *If $\bar{x}$ is an eigenvector corresponding to the minimal eigenvalue of $L + \mathrm{diag}(\bar{\sigma})$ 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 $L + \mathrm{diag}(\bar{\sigma})$ then $\bar{x}$ solves

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

Assume that $\tilde{x}$ solves

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

then

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

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

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

We do not give the proof here but note that the inclusion $\partial f(\bar{\sigma}) \supseteq \mathrm{convhull}(\{x^2 - e;\ x \in \mathscr{E}(\bar{\sigma})\})$ is obvious by Lemma 3.2 and the fact that $\partial f(\bar{\sigma})$ is a convex set.

Corollary 3.4. *Let $\mathscr{E}(\bar{\sigma})$ be the set of all eigenvectors with norm $\sqrt{n+1}$ corresponding to the minimal eigenvalue of $L + \mathrm{diag}(\bar{\sigma})$. Then*

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

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 3.3 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 result.

Corollary 3.5. *The direction d of steepest ascent at σ_0 is given by*

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

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

We will use subgradients in a similar way as gradients are used in a steepest ascent algorithm. Even though there may be many subgradients to choose between, Corollary 3.5 finds the locally best one. Fig. 2 shows the level sets of a function and the subgradients at two different points. To the left, the function is differentiable at σ_0 and hence the only subgradient is the gradient which points in the direction of steepest ascent. To the right, there are several subgradients and the one with the smallest norm points in the direction of steepest ascent.

3.2. Implementation

The basic idea is to find an ascending direction and then to solve an approximation of $f(\sigma)$ along this direction. This process is then repeated until a good solution is found.

3.2.1. Finding ascent directions

The first step is to find an ascending direction. We use Corollary 3.3 to find a good direction. A vector $x \in \mathscr{E}(\bar{\sigma})$ 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 sufficient to pick the vectors $x_i^2 - e$ and use the convex envelope of these vectors as our approximation. Let $\mathscr{S}$ be our approximating set. To determine the best direction, the vector of minimum norm in $\mathscr{S}$ needs to be found. The search can be written as

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

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 3.4 to calculate the directional derivative. In fact, we can solve the optimization problem (14) efficiently by using the parametrization (17), which results in

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

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 $\mathscr{S}$ to improve the approximation. In this way, we either find an ascending direction or we find

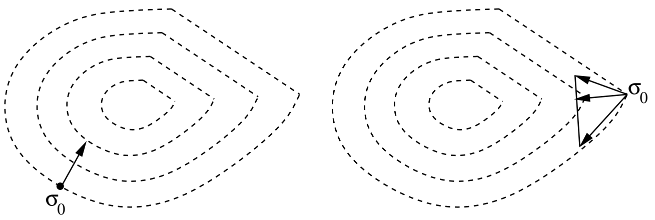


Fig. 2. The level sets of a function and the subgradients at two points. Left: f is differentiable at σ_0 and hence the gradients point in the direction of steepest ascent. Right: f is non differentiable at σ_0 and the direction of steepest ascent is given by the subgradient with the smallest norm.

that zero is a subgradient, meaning that we have reached the global maximum.

3.2.2. Approximating f along a direction

The next step is to find an approximation $\tilde{f}$ of the objective function along a given 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 $L + \mathrm{diag}(\sigma)$. 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

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

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

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

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 does not cross any other eigenvalue. If it does then one can expect that there is some i such that $\inf(f_1(\sigma), f_i(\sigma))$ is a good approximation.

This gives us a function $\bar{f}$ of the type

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

To optimize this function we can solve the linear program

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

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, $f(\tilde{\sigma})$ is greater than $f(\sigma)$, but 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 incorporate these terms to our approximation and repeat the process. Fig. 3 shows two examples of the objective function f and its approximating function $\tilde{f}$.

4. The trust region problem

Another interesting relaxation of our original problem is obtained if we add the additional constraint $y_{n+1} = 1$ to (2). We then obtain the following relaxation:

$$z_{\mathrm{tr}} = \inf_{\|y\|^2 = n} y^{\mathrm{T}}Ay + b^{\mathrm{T}}y. \tag{24}$$

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

$$z_{\mathrm{sp}} \leqslant z_{\mathrm{tr}} \tag{25}$$

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 enforcing the constraint, that is, solving (24) often yields much better solutions. In addition, since the trust region subproblem is a special instance of (6), we have that, given a proper initialization of σ, $z_{\mathrm{sg}} \geqslant z_{\mathrm{tr}}$.

Next, we will show that it is possible to solve (24) exactly. A problem closely related to (24) is

$$\inf_{\|y\|^2 \leqslant n} y^{\mathrm{T}}Ay + b^{\mathrm{T}}y. \tag{26}$$

This problem is usually referred to as the trust region subproblem. Solving the problem is one step in a general optimization scheme for descent minimization and it is known as the trust region method [9]. Instead of minimizing a general function, one approximates it with a second order polynomial $y^T Ay + b^Ty + c$. A constraint of the type $\|y\|^2 \leqslant m$ then specifies the set in which the approximation is believed to be good (the trust region).

The trust region subproblem have been studied extensively in the optimization literature ([21,22,26,25,20]). A remarkable property of this problem is that, even though it is non convex, there is no duality gap (see [3]). In fact, this is always the case when we have a quadratic objective function and only one quadratic constraint. The dual problem of (26) is

$$\sup_{\lambda \leqslant 0} \inf_{y} y^{\mathrm{T}}Ay + b^{\mathrm{T}}y + \lambda(n - y^{\mathrm{T}}y). \tag{27}$$

In [25], it is shown that y^* is the global optimum of (26) if and only if (y^*, λ^*) is feasible in (27) and fulfills the following system of equations:

$$(A - \lambda^* I)y^* = -\frac{1}{2}b \tag{28}$$

$$\lambda^*(n - y^Ty) = 0 \tag{29}$$

$$A - \lambda^* I \succeq 0. \tag{30}$$

The first two equations are the KKT conditions for a local minimum, while the third determines the global minimum. From equation (30) it is easy to see that if A is not positive semidefinite, then λ^* will not be zero. Eq. (29) then tells us that $\|y\|^2 = n$. This shows that for an A that is not positive semidefinite problems (24) and (26) are equivalent. Note that we may always assume that A is not positive semidefinite in (24). This is because we may always subtract mI from A since we have the constant norm condition. Thus replacing

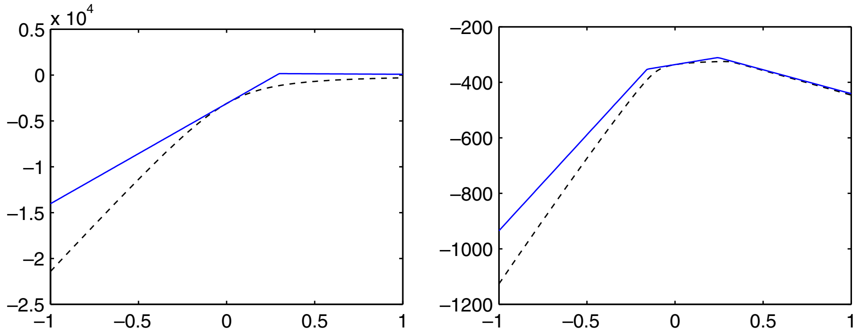


Fig. 3. Two approximations of the objective function $f(\sigma + td)$ along an ascent direction d. The dashed line is the true objective f function and the solid line is the approximation $\tilde{f}$.

A with $A-mI$ for sufficiently large m gives us an equivalent problem with A not positive definite.

A number of methods for solving this problem have been proposed. In [20] semidefinite programming is used to optimize the function $n(\lambda_{min}(H(t))-t)$, where

$$H(t) = \begin{pmatrix} A & \frac{1}{2}b \\ \frac{1}{2}b^T & t \end{pmatrix}, \tag{31}$$

and $\lambda_{\min}$ is the algebraically smallest eigenvalue. In [15] the authors solve $\frac{1}{\psi(\lambda)} - \frac{1}{\sqrt{n}} = 0$ where $\psi(\lambda) = \|(A-\lambda I)^{-1}\frac{1}{2}b\|$. This is a rational function with poles at the eigenvalues of A. To ensure that $A-\lambda I$ is positive semidefinite a Cholesky factorization is computed. If one can afford this, Cholesky factorization is the preferred choice of method. However, the LSTRS-algorithm developed in [21,22] is more efficient for large scale problems. LSTRS works by solving a parametrized eigenvalue problem. It searches for a t such that the eigenvalue problem

$$\begin{pmatrix} A & \frac{1}{2}b \\ \frac{1}{2}b^T & t \end{pmatrix}\begin{pmatrix} y \\ 1 \end{pmatrix} = \lambda_{min}\begin{pmatrix} y \\ 1 \end{pmatrix} \tag{32}$$

or equivalently

$$\begin{aligned} (A-\lambda_{\min}I)y &= -\frac{1}{2}b \\ t-\lambda_{\min} &= -\frac{1}{2}b^{\mathrm{T}}y \end{aligned} \tag{33}$$

has a solution. Finding this t is done by determining a λ such that $\phi'(\lambda) = n$, where ϕ is defined by

$$\phi(\lambda) = \frac{1}{4}b^{\mathrm{T}}(A-\lambda I)^{\dagger}b = -\frac{1}{2}b^{\mathrm{T}}y. \tag{34}$$

It can be shown that λ gives a solution to (33). Since ϕ is a rational function with poles at the eigenvalues of A, it can be expensive to compute. Instead rational interpolation is used to efficiently determine λ. For further details see [21,22].

5. Applications

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 [27] and the spectral bundle algorithm developed by Helmberg [11]. Our spectral subgradient algorithm is implemented in Matlab and the trust region algorithm is based on LSTRS [21] (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).

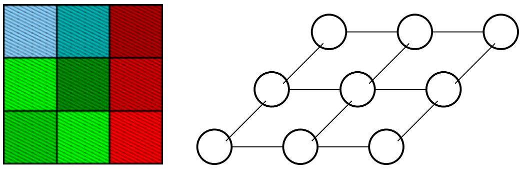


Fig. 4. Graph representation of a 3 × 3 image.

5.1. Segmentation with prior information

In our first example we will compare the trust region method to the spectral relaxation for some image segmentation problems. There is a huge a literature on image based segmentation, see for example, [31,24,4,8,28]. In this section, we will use standard techniques to formulate the problem and describe the performance for different relaxation schemes. We will see that the spectral relaxation (as used in, for example, [24]) can result in poor segmentations when the extra variable y_{n+1} in (2) is not enforced to be ±1. We consider a simple multi-class segmentation problem with prior information.

5.1.1. Graph representations of images

The general approach of constructing an undirected graph from an image is shown in Fig. 4. Basically each pixel in the image is viewed as a node in a graph. Edges are formed between nodes with weights corresponding to how alike two pixels are, given some measure of similarity, as well as the distance between them. In an attempt to reduce the number of edges in the graph, only pixels within a small, predetermined neighborhood $\mathcal{N}$ of each other are considered. Cuts made in such a graph will then correspond to a segmentation of the underlying image.

5.1.2. Including prior information

To be able to include prior information into the visual grouping process we modify the construction of the graphs in the following way. To the graph G we add k artificial nodes. These nodes do not correspond to any pixels in the image, instead they are meant to represent the k different classes into which the image is to be partitioned. The contextual information that we wish to incorporate is modeled by a simple statistical model. Edges between the class nodes and the images nodes are added, with weights proportional to how likely a particular pixel is to a certain class. With the labeling of the k class nodes fixed, a minimal cut on such a graph should group together pixels according to their class likelihood and still preserving the spatial structure, see Fig. 5.

5.1.3. Combinatorial optimization

Next, we show how to approximate this problem using the spectral method and the trust region method.

Let z_{ij} denote the class indicators, that is, $z_{ij} = 1$ if pixel i is in class j and -1 otherwise. Let Z be the $n \times k$ matrix containing the z_{ij} and z_i be the i'th column of Z. If we let W contain the inter-pixel

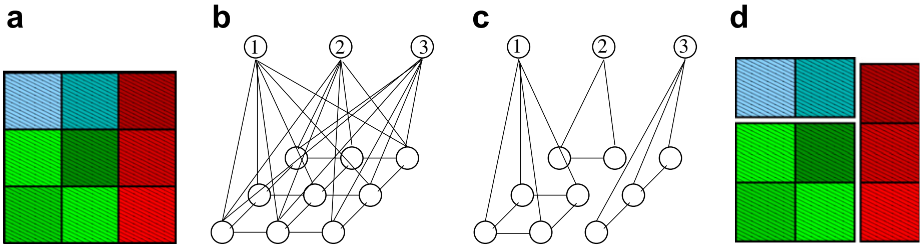


Fig. 5. A graph representation of an image and an example three-class segmentation. Unnumbered nodes corresponds to pixels and numbered ones to the artificial class nodes. (a) Original image, (b) corresponding graph, (c) multi-class min-cut, (d) segmentation.

affinities, the min-cut problem (without pixel class probabilities) can then be written

$$\inf_Z \sum_{i=1}^{k} \sum_{\substack{u \in A_i \\ v \notin A_i}} w_{uv} = \frac{1}{4} \inf_Z \sum_{i,j,l} w_{jl}(z_{ij} - z_{il})^2 = \frac{1}{2} \inf_Z \sum_{i=1}^{k} z_i^T (D - W) z_i. \tag{35}$$

Here, D denotes diag(W e), where e is a vector of all ones. The assignment matrix Z must satisfy $Ze = (2 - k)e$. Eq. (35) only contains information about the spatial structure. The probabilities of a single pixel belonging to a certain class can be included by adding a linear term to (35). We let p_{ij} be the probability that pixel i belongs to class j. If P is the $n \times k$ matrix containing the p_{ij} and p_i is the columns of P. Ignoring the constant 1/2, we get

$$C_{\min} = \inf_{\substack{Z \in \{-1,1\}^{n \times k} \\ Ze = (2-k)e}} \sum_{i=1}^{k} z_i^{\mathrm{T}} (D - W) z_i - \mu p_i^{\mathrm{T}} z_i \tag{36}$$

$$= \inf_{\substack{Z \in \{-1,1\}^{n \times k} \\ Ze = (2-k)e}} \mathrm{tr}\Big(Z^T (D - W) Z\Big) + \mu \mathrm{tr}\Big(P^T Z\Big). \tag{37}$$

Here, μ is a parameter controlling the balance between the quadratic smoothing term and the linear context term. The first term in (37) is quadratic and the second term is linear. By reshaping Z to a vector z (by column stacking Z) we can write the objective function as a quadratic form $z^TA\,z + b^Tz$. Further more we write the linear constraints $Ze = (2 - k)e$ as $Cz = d$. Since $z \in \{-1,1\}^{nk} \Rightarrow \|z\|^2 = nk$, we replace the binary constraints with the norm constraint and obtain the relaxation

$$\inf_{\|z\|^2 = nk} z^T A z + b^T z \tag{38}$$

$$Cz = d. \tag{39}$$

This is almost a trust region problem, however, in addition we also have the constraints $Cz = d$. Next we will show how to eliminate these constraints. Note that this procedure is not specific for the $Ze = (2 - k)e$ constraints but can be applied to any type of linear equality constraint system. The space of solutions to $Cz = d$ can be parametrized as $z = Qy + v$, where Q and v can be chosen so that $Q^TQ = I$ and $Q^Tv = 0$. Here, $y \in \mathbb{R}^{nk-r}$ where r is the number of rows in C. With this change of coordinates we arrive at the following quadratically constrained quadratic program

$$\begin{aligned} &\inf_y (Qy + v)^{\mathrm{T}} A (Qy + v) + 2b^{\mathrm{T}}(Qy + v) \\ &\text{s.t.} (Qy + v)^{\mathrm{T}}(Qy + v) = nk. \end{aligned} \tag{40}$$

For efficiently solving this problem we here turn our attention to two relaxations that are tractable from a computational perspective. Letting $\tilde{A} = Q^T A Q$, $\tilde{b} = 2Q^T A^T v + Q^T b$ and dropping the constant $b^{\mathrm{T}}v + v^{\mathrm{T}}\,Av$, we obtain the equivalent trust region problem

$$\mu_{\mathrm{tr}} = \inf_{\|y\|^2 = nk - v^{\mathrm{T}} v} y^{\mathrm{T}} \tilde{A} y + 2\tilde{b}^{\mathrm{T}} y. \tag{41}$$

By adding an extra variable $y_{n(k-1)+1}$ as in (2) we obtain the spectral relaxation.

5.1.4. Experimental results

As mentioned in the previous section prior knowledge is incorporated into the graph cut framework through the k artificial nodes. For this purpose we need a way to describe each pixel as well as model the probability of that pixel belonging to a certain class.

The image descriptor in the current implementation is based on color alone. Each pixel is simply represented by their three RGB color channels. The probability distribution for these descriptors are modeled using a Gaussian Mixture Model (GMM).

$$p(v_{RGB}|\Sigma, \mu) = \sum_i \frac{1}{\sqrt{2\pi|\Sigma_i|}} e^{\left(-\frac{1}{2}(v_{RGB} - \mu_i)^T \Sigma_i^{-1} (v_{RGB} - \mu_i)\right)}. \tag{42}$$

Here, v_{RGB} is the RGB value of the pixel, μ_i and Σ_i is the mean and the covariance matrices for the ith Gaussian, respectively. From a number of manually annotated training images the GMM parameters are then fitted through Expectation Maximization, [6]. This fitting is only carried out once and can be viewed as the learning phase of our proposed method.

The edge weight between pixel i and j and the weights between pixel i and the different class-nodes are given by

$$w_{ij} = e^{\left(-\frac{r(i,j)}{\sigma_R}\right)} e^{\left(-\frac{\|v_{RGB}(i) - v_{RGB}(j)\|^2}{\sigma_W}\right)} \tag{43}$$

$$p_{ki} = \alpha \frac{p(v_{RGB}(i)|i \in k)}{\sum_j p(v_{RGB}(i)|i \in j)}. \tag{44}$$

Here, $\|\cdot\|$ denotes the euclidian norm, $r(i,j)$ the distance between pixel i and j. The tuning parameters σ_R and σ_W weight the importance of the different features. Hence, w_{ij} contains the inter-pixel similarity that ensures that the segmentation is coherent. The number p_{ki} describes how likely pixel i is to belong to class k and α is a parameter weighting the importance of spatial structure vs. class probability.

Preliminary tests of the suggested approach were carried out on a limited number of images. We chose to segment the images into four common classes: sky, grass, brick and background. Gaussian mixture models for each of these classes were firstly acquired from a handful of training images manually chosen as being representative of such image regions, see Fig. 6.

For an unseen image the pixel affinity matrix W and class probabilities were computed according to (43) and (44). The resulting optimization program was then solved using both the spectral relaxation (SR) and the trust region subproblem (TSP) method. The outcome can be seen in Fig. 7. Parameters used in these experiments were $\sigma_R = 1$, $\sigma_W = 1$, $\alpha = 10$ and $\mathcal{N}$ a 9×9 neighborhood structure.

Both relaxations produce visually relevant segmentations. Even though our approach is based on very limited training data, it does appear to use the prior information in a meaningful way. Taking a closer look at the solutions supplied by the trust region method and the spectral relaxation for these two examples, one substantial difference is revealed. The spectral relaxation was reached by ignoring the constraint on the homogenized coordinate $y_{n(k-1)+1} = 1$. The solutions to the examples in Fig. 7 produce a homogeneous coordinate value of $y_{n(k-1)+1} \approx 120$ in both cases. As the class probabilities of the pixels are represented by the linear part of (39), the spectral relaxation yields an image partition that weights prior information much higher than spatial coherence. Any spatial structure of an image will thus not be preserved. The spectral relaxation is basically just a maximum-likelihood classification of each pixel individually.

5.2. Binary restoration

Next we will evaluate the performance of the subgradient algorithm. As a test problem, we consider the problem of separating a signal from noise, similar to what is done in [12]. The signal $\{x_i\}$, $i = 1,\ldots,n$, 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 $\{s_i\}$, $i = 1,\ldots,n$. Figs. 8(a) and (b) graph 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:

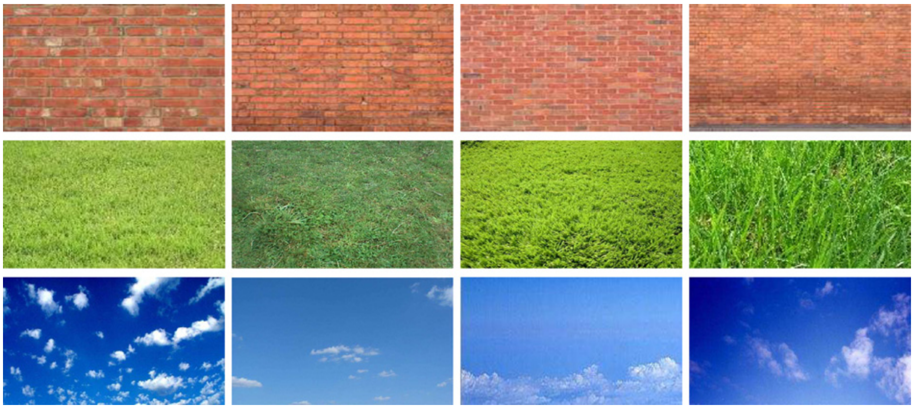
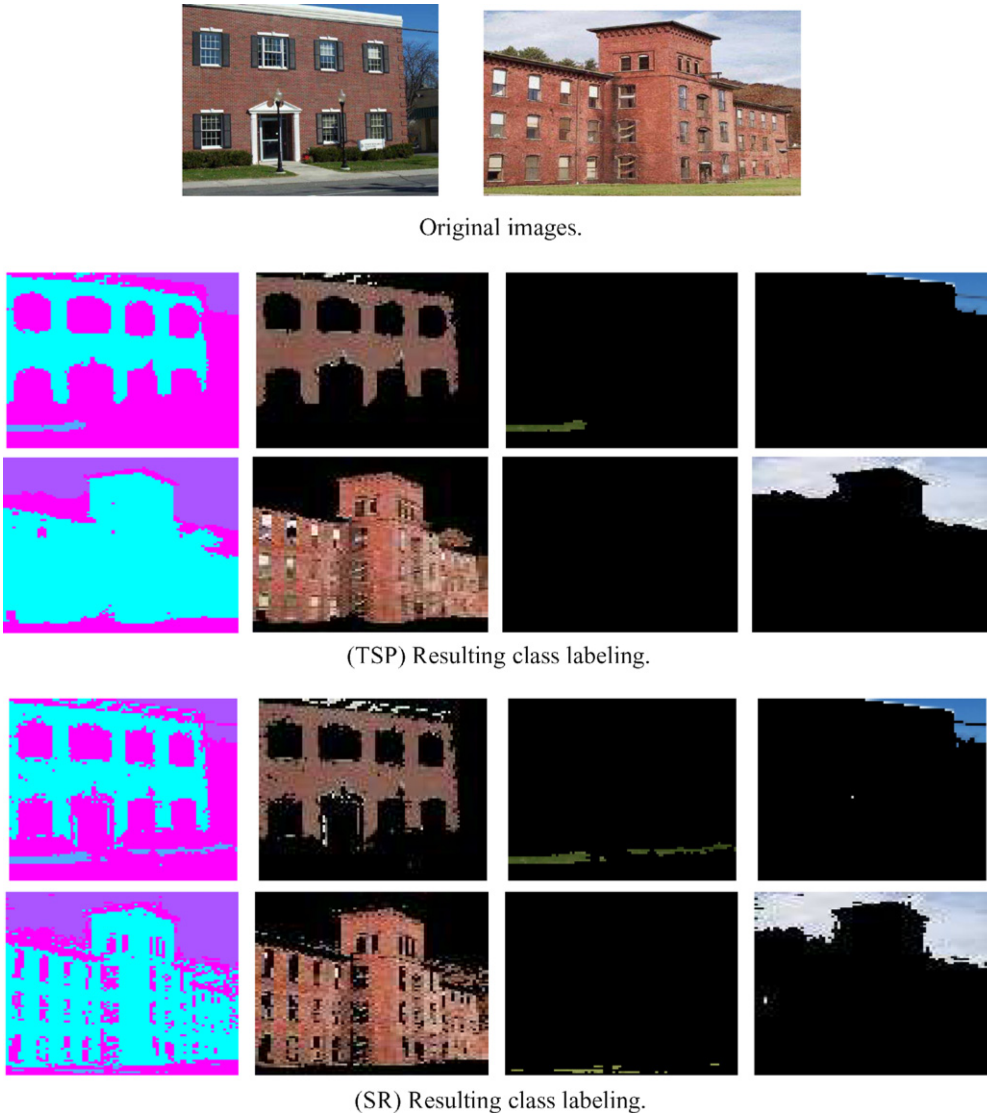


Fig. 6. Sample training images.



Original images.

(TSP) Resulting class labeling.

(SR) Resulting class labeling.

Fig. 7. Example segmentation/classification of an image using both Trust Region Subproblem (TSP) formulation and Spectral Relaxation (SR).

$$\sum_i (x_i - s_i)^2 + \mu \sum_i \sum_{j \in N(i)} (x_i - x_j)^2, \quad x_i \in \{-1, 1\}. \tag{45}$$

Here, $N(i)$ means a neighborhood of i, in this case $\{i - 1, i + 1\}$. The reason for using this test problem is the global optimum can be computed efficiently using dynamic programming.

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. Note that only one problem instance has been generated for each n. For comparison we have also computed the global opti-

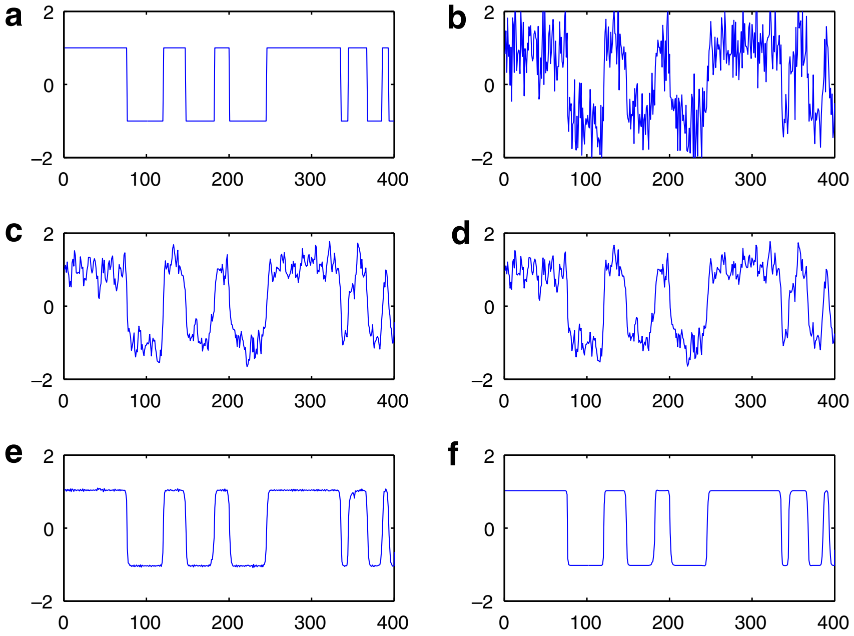


Fig. 8. Computed solutions for the signal problem with $n = 400$. (a) Original signal, (b) signal + noise, (c) solution obtained using spectral relaxations, (d) trust region, (e) subgradient algorithm and (f) dual semidefinite program.

mum using dynamic programming. For the subgradient method, 10 iterations were run and in each iteration, the 15 smallest eigenvectors were computed for the approximation set $\mathscr{S}$ in (18). Note in particular the growth rate of the execution times for the SDP. Figs. 8(b)–(d) show 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 value of x_{n+1} happens to be close to ±1.

Next, we consider a similar problem as above, which was also a test problem in [12]. We want to restore the map of Iceland given in Fig. 9. The objective function is the same as in (45), 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 [12], a different SDP solver was used and the execution time was 64885 s. Instead, we compare with the spectral bundle algorithm [11]. Table 3 gives the execution times and the objective values of the estimates. Fig. 10 shows the resulting restorations for the different methods.

For the subgradient algorithm, the four 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 Fig. 10(d) we had to replace these pixel values with a smaller value. This results in the white areas in Fig. 10(d) and the bar close to the value 2 in Fig. 10(d).

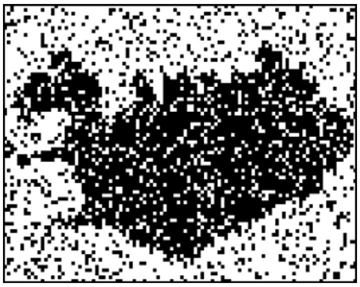


Fig. 9. Map of Iceland corrupted by noise.

Table 1
Execution times in seconds for the signal problem

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 2
Objective values of the relaxations

n	Spectral	Trust region	Subgradient	SDP	DP
100	24.3	31.6	40.6	53.1	53.5
200	27.4	40.5	53.5	76.1	76.2
400	74.9	88.4	139	174	184
600	134	164	240	309	322
800	169	207	282	373	383
1000	178	229	322	439	451

A higher value means a better lower bound for the optimal value. The last column is the optimal value obtained by dynamic programming.

5.3. Partitioning

In this section, we consider the problem of partitioning an image into perceptionally different parts, cf. [12]. Fig. 11(a) shows the image that is to be partitioned. Here we want to separate the

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

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

buildings from the sky. To do this we use the following regularization term

$$\sum_{ij} w_{ij}(x_i - x_j)^2, \tag{46}$$

where the weights are according to (43). To avoid solutions where all pixels are put in the same partition, and to favor balanced partitions, a term penalizing unbalanced solutions is added. If one adds the constraint $e^{\mathrm{T}}\ x = 0$ or equivalently $x^{\mathrm{T}}\ ee^{T}x = 0$ 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

$$\inf x^T(L + \mu ee^{\mathrm{T}})x,\ x_i \in \{-1, 1\}. \tag{47}$$

Observe that this problem is not submodular [13]. Since the size of the skyline image (Fig. 11(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 $(L + \mu ee^{\mathrm{T}})x$ 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. Fig. 11(b) shows the resulting partition. Figs. 11(e) and (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 partitions obtained from Normalized Cuts (NC) [24] which is a frequently applied method for segmentation. The reason for the strange partitioning in Figs. 11(c) and (d) is that the obtained NC solution essentially only 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 $(L + \mu ee^{T})$.

5.4. 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 [2,23,29].

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

$$\sum w_{ijkl}x_{ij}x_{kl}, \tag{48}$$

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,

$$\mathrm{abs}(\|s_i - s_k\| - \|t_j - t_l\|) < \theta, \tag{49}$$

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

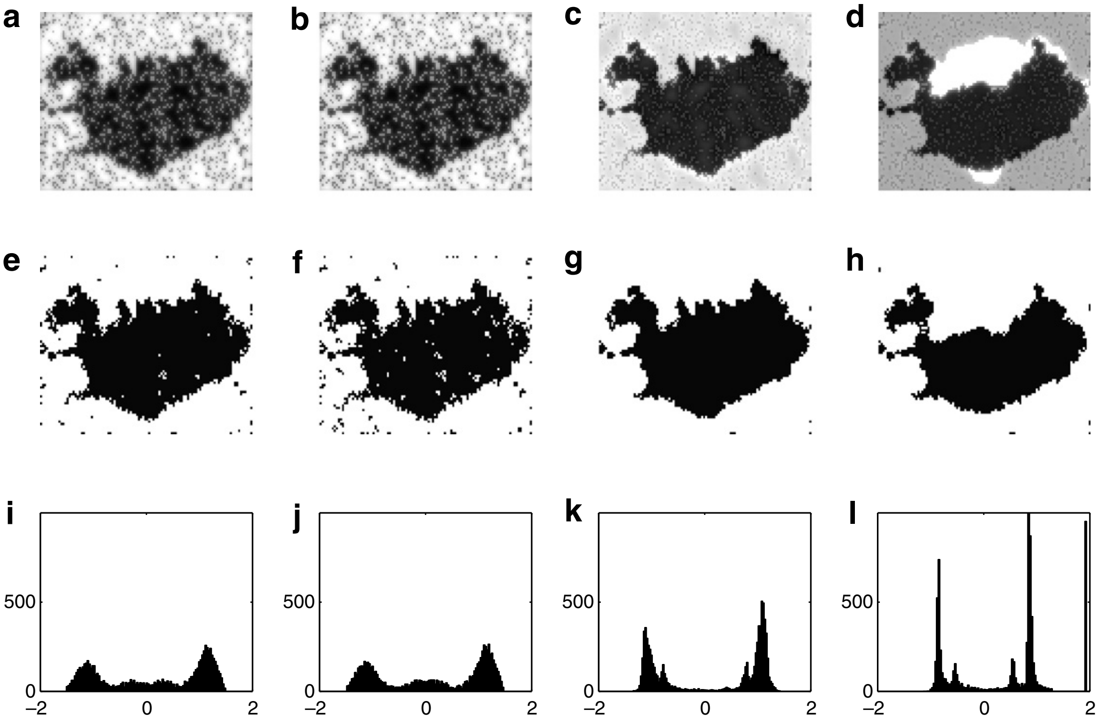


Fig. 10. Top row: relaxed solutions. Middle: thresholded solutions. Bottom: histogram of the estimated pixel values. (a,e,i): spectral method, (b,f,j): trust region, (c,g,k): subgradient, 10 iterations, (d,h,l): Helmberg's bundle method, 5 iterations.

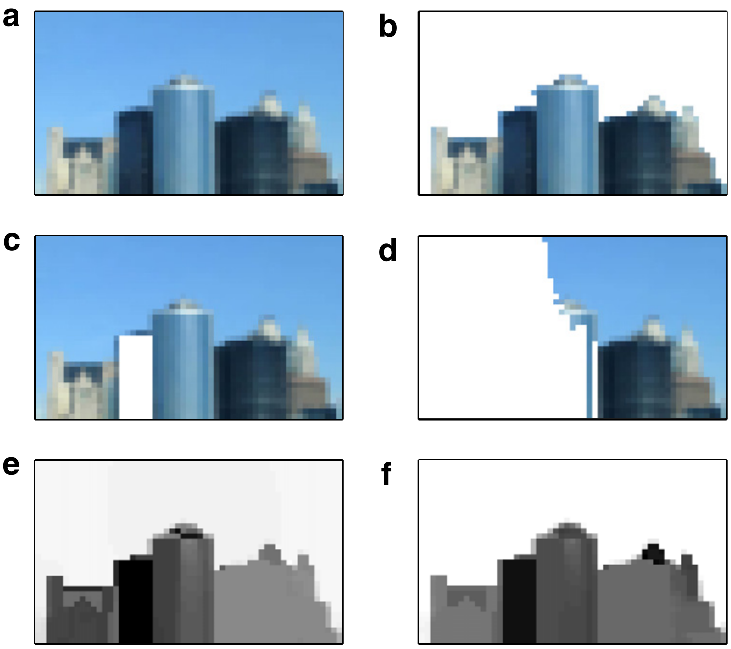


Fig. 11. (a) Original image, (b) thresholded segmentation with 7 iterations of the subgradient algorithm (white pixels correspond to one class, remaining pixels are in the other class), (c) NC solution thresholded at the median, (d) NC solution thresholded at the mean, (e,f) relaxed (untruncated) solutions obtained with 4 and 7 iterations, respectively, of the subgradient algorithm.

Table 4
Computing times for the skyline image

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

$j = 1,\ldots,n$ 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 do 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, added noise and applied a random Euclidean motion. Figs. 12(a) and (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 Fig. 13 and the resulting registration for the subgradient method is shown in Fig. 14. The standard spectral relaxation for this problem works rather poorly 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 [23,29]. For more information on the quality of SDP relaxations for this problem, the reader is also referred to the same papers.

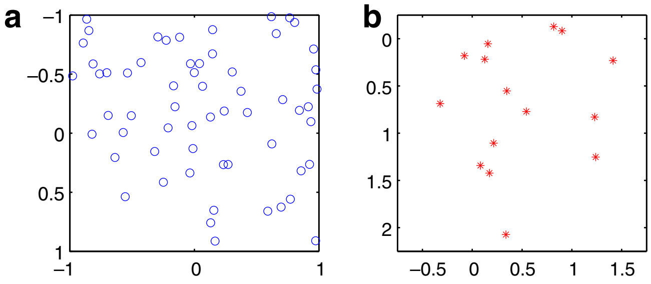


Fig. 12. One random example for the registration problem: (a) Target points $n = 60$ and (b) source points $m = 15$.

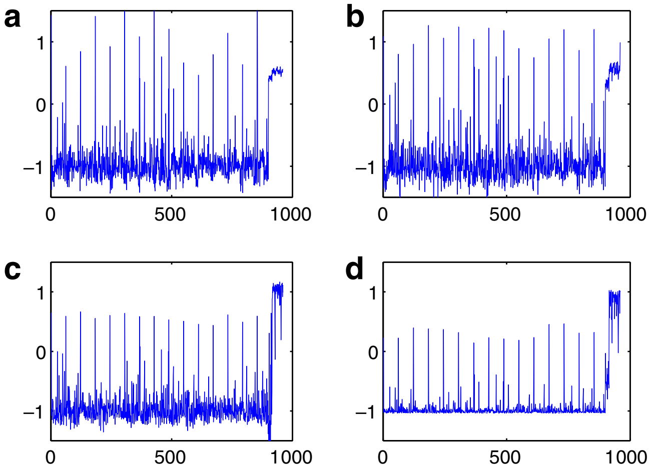


Fig. 13. Computed solutions $z = [z_{11}, z_{12}, \ldots, z_{m+1,n}]$ for the registration problem using (a) the trust region method, (b) the subgradient method, 7 iterations, (c) the subgradient method, 15 iterations, and (d) SDP with SeDuMi, cf. Fig. 12.

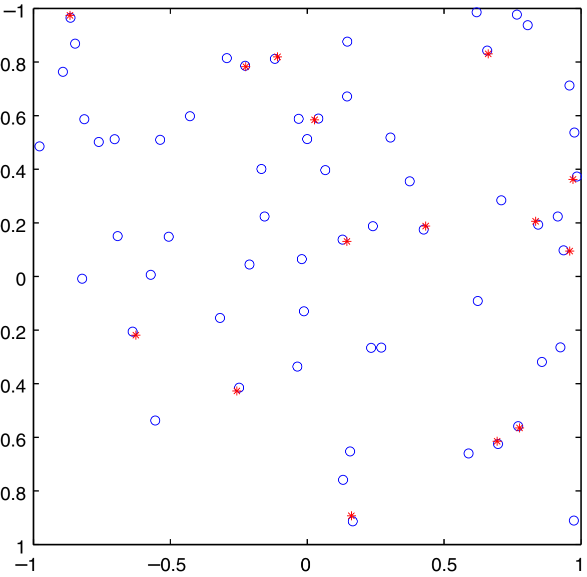


Fig. 14. Registration of the source points to their corresponding target points, cf. Fig. 12.

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

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

6. 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] M.S. Bazaraa, H.D. Sherali, C.M. Shetty, Nonlinear Programming, Theory and Algorithms, Wiley, 1993.
[2] A.C. Berg, T.L. Berg, J. Malik, Shape matching and object recognition using low distortion correspondences, in: Conf. Computer Vision and Pattern Recognition, San Diego, USA, 2005, pp. 26–33.
[3] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[4] Y. Boykov, O. Veksler, R. Zabih, Fast approximate energy minimization via graph cuts, IEEE Trans. Pattern Anal. Mach. Intell. 23 (11) (2001) 1222–1239.
[5] T. Cour, J. Shi, Solving markov random fields with spectral relaxation, in: Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics, vol. 11, 2007.
[6] A. Dempster, M. Laird, D. Rubin, Maximum likelihood from incomplete data via the em algorithm, JR Stat. Soc. (1977).
[7] A.P. Eriksson, C. Olsson, F. Kahl, Image segmentation with context, in: Proc. Scandinavian Conference on Image Analysis, Aalborg, Denmark, 2007.
[8] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient graph-based image segmentation, Int. J. Comput. Vis. 59 (2) (2004) 167–181.
[9] R. Fletcher, Practical Methods of Optimization, John WIley & Sons, 1987.
[10] M.X. Goemans, D.P. Wiliamson, Improved approximation algorithms for maximum cut and satisfiability problem using semidefinite programming, J. ACM 42 (6) (1995) 1115–1145.
[11] C. Helmberg, F. Rendl, A spectral bundle method for semidefinite programming, SIAM J. Optim. 10 (3) (2000) 673–696.
[12] J. Keuchel, C. Schnörr, C. Schellewald, D. Cremers, Binary partitioning, perceptual grouping, and restoration with semidefinite programming, IEEE Trans. Pattern Anal. Mach. Intell. 25 (11) (2003) 1364–1379.
[13] V. Kolmogorov, R. Zabih, What energy functions can be minimized via graph cuts?, IEEE Trans Pattern Anal. Mach. Intell. 26 (2) (2004) 147–159.
[14] A. Levin, A. Rav-Acha, D. Lischinski, Spectral matting, in: Proc. Conf. Computer Vision and Pattern Recognition, Minneapolis, USA, 2007.
[15] J.J. Moré, D.C. Sorensen, Computing a trust region step, SIAM J. Sci. Stat. Comput. 4 (3) (1983) 553–572.
[16] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: Advances in Neural Information Processing Systems, vol. 14, 2002.
[17] C. Olsson, A.P. Eriksson, F. Kahl, Solving large scale binary quadratic problems: Spectral methods vs. semidefinite programming, in: Proc. Conf. Computer Vision and Pattern Recognition, Minneapolis, USA, 2007.
[18] J. Park, H. Zha, R. Kasturi, Spectral clustering for robust motion segmentation, in: European Conf. Computer Vision, Prague, Czech Republic, 2004.
[19] S. Poljak, F. Rendl, H. Wolkowicz, A recipe for semidefinite relaxation for (0,1)-quadratic programming, J. Global Optim. 7 (1995) 51–73.
[20] F. Rendl, H. Wolkowicz, A semidefinite framework for trust region subproblems with applications to large scale minimization, Math. Prog. 77 (2 Ser.B) (1997) 273–299.
[21] M. Rojas, S.A. Santos, D.C. Sorensen, A new matrix-free algorithm for the large-scale trust-region subproblem, SIAM J. Optim. 11 (3) (2000) 611–646.
[22] M. Rojas, S.A. Santos, D.C. Sorensen, Lstrs: Matlab software for large-scale trust-region subproblems and regularization. Technical Report 2003-4, Department of Mathematics, Wake Forest University, 2003.
[23] C. Schellewald, C. Schnörr, Probabilistic subgraph matching based on convex relaxation, in: Proc. Int. Conf. on Energy Minimization Methods in Computer Vision and Pattern Recognition, 2005, pp. 171–186.
[24] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[25] D.C. Sorensen, Newton's method with a model trust region modification, SIAM J. Numerical Anal. 19 (2) (1982) 409–426.
[26] D. C Sorensen, Minimization of a large-scale quadratic function subject to a spherical constraint, SIAM J. Optim. 7 (1) (1997) 141–161.
[27] J.F. Sturm, Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones, Optim. Methods Software 11–12 (1999) 625–653.
[28] D. Tolliver, G. Miller, Graph partitioning by spectral rounding: Applications in image segmentation and clustering, in: Conf. Computer Vision and Pattern Recognition, New York City, USA, 2006, pp. 1053–1060.
[29] P.H.S. Torr, Solving markov random fields using semi definite programming, in: Ninth International Workshop on Artificial Intelligence and Statistics, 2003.
[30] S. Umeyama, An eigendecomposition approach to weighted graph matching problems, IEEE Trans. Pattern Anal. Mach. Intell. 10 (5) (1988) 695–703.
[31] Y. Weiss, Segmentation using eigenvectors: a unifying view, in: Int. Conf. Computer Vision, Kerkyra, Greece, 1999, pp. 975–982.