Descent heuristics for unconstrained minimization

Florian Jarre, Ania López

University of Düsseldorf, Germany

Abstract

Semidefinite relaxations often provide excellent starting points for nonconvex problems with multiple local minimizers. This work aims to find a local minimizer within a certain neighborhood of the starting point and with a small objective value. Several approaches are motivated and compared with each other.

Key words. Descent method, unconstrained minimization, local minimizer.

1 Introduction

Let $f : \mathbb{R}^n \to \mathbb{R}$ be three times continuously differentiable. The problem under consideration is to find – among several local minimizers of f – a local minimizer x^* with a small value $f(x^*)$. We compare several heuristics that do not guarantee to find a global minimizer but that aim to find a local minimizer with "rather small value of f".

For certain problems with multiple local solutions semidefinite programs often provide excellent starting points. Examples are sensor location problems, see e.g. [3], or low-rank formulations of combinatorial problems. The goal of this article is to find "good" local minimizers within a certain domain of attraction given by the initial point. We emphasize that this can only be a heuristic; nevertheless, numerical examples indicate that the new approaches proposed here often lead to better local solutions than the steepest descent path.

For smooth strictly convex functions f and under standard conditions (regarding the descent) all descent paths eventually lead to the same optimal solution, and Newton's method generally performs very well. Our aim is not to compete with Newton's method but to provide a cheap first order method that is "efficient" *far away* from a local solution when the Hessian of f is not positive definite. Nearby a solution first order methods – such as the ones proposed here – are poor and generally need to be replaced with other approaches.

The following principles for minimizing f starting at a given point $x^0 \in \mathbb{R}^n$ motivate the new approaches:

1. In order to remain within a domain of attraction given by x^0, only descent steps are considered.

2. As the global behavior – far from a local minimizer – is of interest, only short steps derived from first order approximations are considered.

3. The first order predictions used in this paper are based on the following naive motivation for *nonconvex* minimization:

   (a) Aim towards points where the gradient gets larger (and f gets smaller) because large gradients hint that a large descent may be possible from there.

(b) Aim towards points where the negative curvature is amplified (and f gets smaller) because negative curvature hints that long descent steps may be possible from there.

We repeat that both motivations make little sense for convex minimization.

Below, the following notation is used.

- By $A \bullet B = \sum_{i,j} A_{i,j} B_{i,j} = \text{trace}(A^T B)$ we denote the scalar product of two matrices.
- We will call $g = \nabla f(x)$ the gradient and $H = \nabla^2 f(x)$ the Hessian of f at the point x.
- The orthogonal eigenvalue decomposition of H is given by $H = UDU^T$, i.e. U is a unitary matrix whose columns u^i are eigenvectors to the eigenvalues $D_{ii} = \lambda_i(H) = \lambda_i$. Without loss of generality we assume that the eigenvalues are sorted in decreasing order and the first p are positive (i.e. $\lambda_1 \geq ... \geq \lambda_p \geq 0 > \lambda_{p+1} \geq ... \geq \lambda_n$, $p = 0$ and $p = n$ is, of course, possible).

2 Increasing gradient descent

Given an iterate x^k, the progress of a short step along the steepest descent direction is proportional to the norm of $\nabla f(x^k)$. To be able to have a large descent in the next step, it may therefore be of advantage not only to minimize f but also to maximize $\|\nabla f\|$ at the same time. This idea leads to a descent step for the function

$$\varphi(x, \alpha) := f(x) - \alpha \|\nabla f(x)\|^2$$

where $\alpha > 0$ is a parameter that balances the reduction of f versus the increase of $\|\nabla f\|$.

The steepest descent direction for φ is given by

$$\Delta x = -\nabla f(x^k) + 2\alpha \nabla^2 f(x^k) \nabla f(x^k).$$

We stress that matrix vector products such as $\nabla^2 f(x^k) v$ can typically be computed (for example by automatic differentiation) cheaply without forming $\nabla^2 f(x^k)$.

The search step is given by the partial derivative $\Delta x = -\nabla_x \varphi(x, \alpha)$ where the choice of $\alpha = \alpha(x)$ depends on x.

Let

$$d^1 := -\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|} \quad \text{and} \quad d^2 := \frac{\nabla(\|\nabla f(x^k)\|^2)}{\|\nabla(\|\nabla f(x^k)\|^2)\|} = \frac{\nabla^2 f(x^k) \nabla f(x^k)}{\|\nabla^2 f(x^k) \nabla f(x^k)\|}$$

be the normalized steepest descent directions for f and for $-\|\nabla f\|_2^2$. Any Δx with $\Delta x^T d^1 < 0$ and $\Delta x^T d^2 < 0$ is a descent direction for both, f and $-\|\nabla f\|_2^2$. The normalized descent direction $\widetilde{\Delta x} := \Delta x / \|\Delta x\|$ is chosen as a nonnegative combination of the vectors d^1 and d^2, i.e.

$$\widetilde{\Delta x} = \beta d^1 + \gamma d^2$$

where $\beta \geq 0$ and $\gamma \geq 0$ depend on α.

A normalized measure of descent for f is given by $\widetilde{\Delta x}^T d^1$ and ranges from 1 (for $\beta = 1,\ \gamma = 0$) to $(d^1)^T d^2$ (for $\beta = 0,\ \gamma = 1$), a negative value of $\widetilde{\Delta x}^T d^1$ indicating ascent.

To find a meaningful choice of α, or, equivalently of β and γ subject to the condition

$$1 = \|\widetilde{\Delta x}\|^2 = \beta^2 + 2\beta\gamma(d^1)^T d^2 + \gamma^2$$

we require that

$$\widetilde{\Delta x}^T d^1 = \rho + (1-\rho)(d^1)^T d^2$$

for some $\rho \in (0, 1]$. The number ρ quantifies the amount of descent of f compared to the steepest descent. "$\rho = 1$" is the plain steepest descent direction, "$\rho = 0$" is the steepest descent direction for $\|\nabla f\|$ and could, in the "worst case", be opposite to $-\nabla f$. When $\rho \geq 0.5$ it follows that $\widetilde{\Delta x}^T d^1 \geq 0$.

Let

$$c := (d^1)^T d^2 = -\frac{\nabla f(x^k)^T D^2 f(x^k) \nabla f(x^k)}{\|\nabla f(x^k)\| \|D^2 f(x^k) \nabla f(x^k)\|}$$

be the cosine of the angle $\sphericalangle(d^1, d^2)$. The numbers β, γ then solve the system of equations

$$\beta^2 + 2\beta\gamma c + \gamma^2 = 1$$

$$\beta + \gamma c = r := \rho + (1-\rho)c.$$

The solution is given by

$$\gamma = \sqrt{\frac{1-r^2}{1-c^2}} = \sqrt{\frac{(1-\rho)(1+\rho+c-\rho c)}{1+c}},$$

and $\beta = \rho + c(1-\rho-\gamma)$.

For the interesting case that the increase of $\|\nabla f\|$ is given high priority and the descent property is barely guaranteed, i.e. for $\rho = 0.5$ the expression for γ simplifies to

$$\gamma = \sqrt{\frac{3+c}{4+4c}}.$$

The choice of β, γ can be translated back to the choice of α: Multiplying the equation

$$\beta d^1 + \gamma d^2 = \frac{\Delta x}{\|\Delta x\|} = \frac{-\nabla f(x^k) + 2\alpha D^2 f(x^k) \nabla f(x^k)}{\| - \nabla f(x^k) + 2\alpha D^2 f(x^k) \nabla f(x^k)\|}$$

from left with $(\beta d^1 + \gamma d^2)^T$, using the definitions of d^1, d^2, and squaring the result we obtain

$$1 = \frac{(\|\nabla f(x^k)\|(\beta + \gamma c) + 2\alpha \|D^2 f(x^k) \nabla f(x^k)\|(\gamma + \beta c))^2}{\| - \nabla f(x^k) + 2\alpha D^2 f(x^k) \nabla f(x^k)\|^2}.$$

Solving this quadratic equation for α yields

$$\alpha = \alpha(x^k) = \frac{\gamma}{2\beta} \frac{\|\nabla f(x^k)\|}{\|D^2 f(x^k) \nabla f(x^k)\|}.$$

A descent method for φ based on this choice of α for a given parameter $\rho \in (0,1]$ is called "increasing gradient descent method with parameter ρ" (briefly "$igd(\rho)$").

For further notation we will denote the given search direction for the iterate x^k by $\Delta x^k_{igd(\rho)} (= \beta d^1 + \gamma d^2)$ as it depends only on ρ (and x^k).

2.1 First Example

The behavior of the descent path $igd(\rho)$ with $\rho = 1$ (steepest descent) and $\rho = 0.6$ is illustrated with a simple example: For $x \in I\!\!R^2$ let

$$f(x) := 10x_1 + x_2 + x_1^2(4 + \frac{x_2^2}{4}). \quad (1)$$

The function f is unbounded below (it tends to $-\infty$ along the line $x_1 = 0,\ x_2 < 0$) and has a local minimizer at $x_1 = -1,\ x_2 = -2$. Figures 1 and 2 below indicate the starting points for which the steepest descent method ($igd(1)$) and $igd(0.6)$ converge to the local minimizer. For all other starting points the methods converge to the infimum of f (i.e. to $-\infty$).

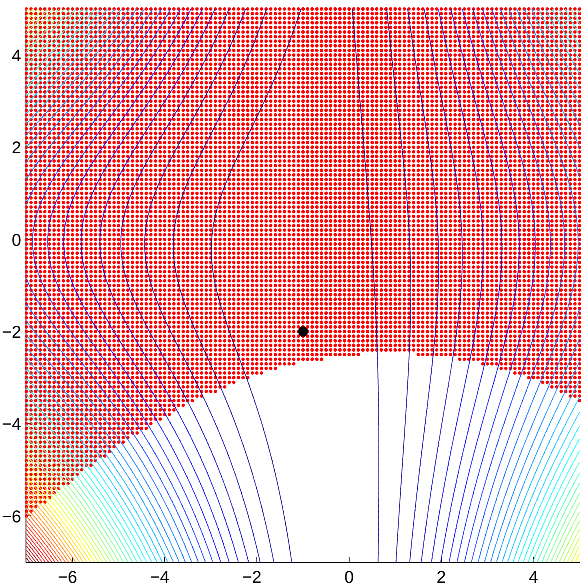

Figure 1: Area indicating starting points from which $igd(1)$ convergences to the (poor) local minimizer of f defined in (1).

The dark (red) shaded areas mark the set of starting points from which $igd(\rho)$ converges to the "poor" local minimizer $(-1,-2)^T$ missing the points $(0,x_2)^T$ for $x_2 \to -\infty$ with lower objective value. For $\rho = 0.6$ this area is much smaller than for $\rho = 1$ (steepest descent). Figure 1 also includes the level sets of f. As expected, the contour lines are perpendicular to the boundary of the

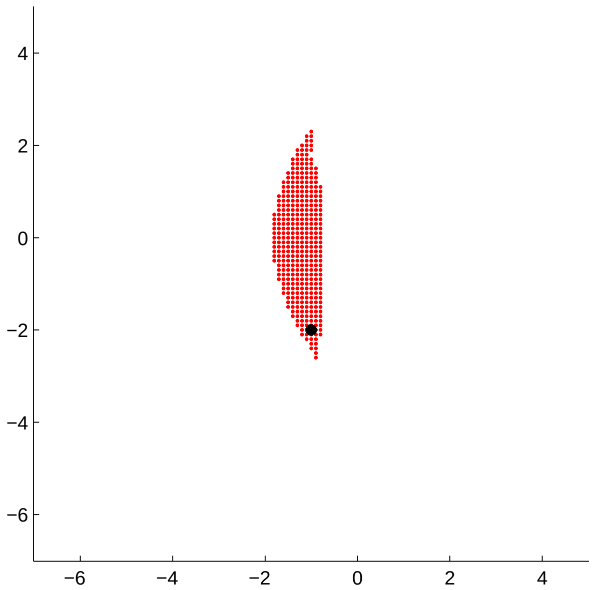

Figure 2: Area indicating starting points from which $igd(0.6)$ convergences to the (poor) local minimizer of f defined in (1).

dark shaded area. We do not include the level sets of $\varphi(x, \alpha(x))$. In fact, (for $\rho < 1$) the $igd(\rho)$-steps "$-\nabla_x \varphi(x, \alpha(x))$" are not necessarily perpendicular to the level sets of φ since α is not constant in general.

2.2 Second Example

For illustration we also display the plot of a steepest descent path and the $igd(0.6)$-path for a convex quadratic function f, $f(x) = \frac{1}{2}x_1^2 + x_2^2$.

In this example, the norm of the gradient is increased at first while f is reduced. For a function f that is bounded below, the decrease of both, $-\|\nabla f\|^2$ and f must stop at some point y. When reaching the point y, the direction of increasing gradient points opposite to the descent direction and is thus ignored.

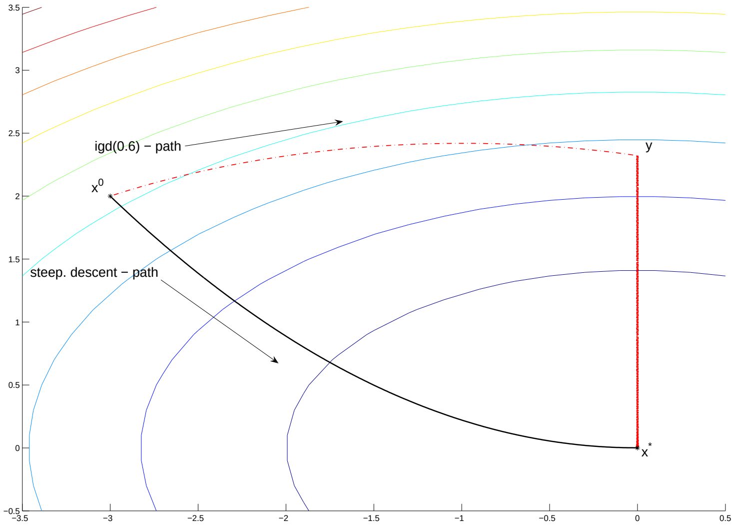

Figure 3: Red, dashed: path of $igd(0.6)$, black: path of steepest descent

3 Curvature descent

In this section and the next one, the descent path is modified based on the use of second and third order derivatives.

Given an iterate x, let $\tilde{g} := U^T g$ (given by the gradient g of f at x and the eigenvalue decomposition of the Hessian $H = UDU^T$) . Without loss of generality we assume that the components $\tilde{g}_i$ of $\tilde{g}$ satisfy

$$\tilde{g}_i \leq 0.$$

(Else multiply the corresponding column u^i of U by -1.) For small $\epsilon > 0$ it follows that

$$f(x + \epsilon u^i) \approx f(x) + \epsilon g^T u^i = f(x) + \epsilon \tilde{g}^T U^T u^i = f(x) + \epsilon \tilde{g}_i,$$

and

$$\|\nabla f(x + \epsilon u^i)\|^2 \approx \|\nabla f(x)\|^2 + 2\epsilon g^T H u^i = \|\nabla f(x)\|^2 + 2\epsilon \tilde{g}_i D_{ii}.$$

Thus, when $D_{ii} < 0$ (and $\tilde{g}_i \neq 0$) a selection of $\omega_i > 0$ effects that both, f is locally reduced along $x + \omega_i u^i$, and $\|\nabla f\|^2$ is increased. On the other hand, for $D_{ii} > 0$ an increase of $\|\nabla f\|^2$ along $\omega_i u^i$ is possible only at the expense of increasing f at the same time. The concept below avoids such search directions u^i as far as possible.

Any search direction $\Delta x \in I\!R^n$ can be written as $\Delta x = \sum_{i=1}^n \omega_i u^i$ with $\omega_i \in I\!R$. We set

$$\tilde{D}_{ii} := \max\{\varepsilon, D_{ii}\}$$

for a small value $\varepsilon > 0$ so that the positive definite matrix $\tilde{H} = U\tilde{D}U^T$ defines a descent direction

$$\Delta x_{cd} = -\tilde{H}^{-1}g$$

for f. The search step Δx_{cd} depends on the choice of $\varepsilon > 0$. As $\varepsilon \to 0$ it follows that $\Delta x_{cd}/\|\Delta x_{cd}\|$ converges to the space

$$Q := span\{u^i \mid D_{ii} < 0\}.$$

For small $\varepsilon > 0$ the descent step Δx_{cd} lies in some sense near the space Q of negative curvature. It is therefore called curvature descent step.

In numerical examples small positive values of ε produced slightly better local minimizers than the limiting direction $\Delta x_{cd}/\|\Delta x_{cd}\|$ obtained for $\varepsilon \to 0$.

4 Decreasing curvature descent

The approach in sections 2 and 3 was motivated by aiming towards points from which a large descent can be anticipated since the norm of the gradient is increasing. This approach can be generalized aiming at areas with "more negative curvature". At points with negative curvature also a long way down to the next minimizer can be anticipated. Both anticipations are purely heurisitcal, but nevertheless numerical experiments indicate a good performance.

Let

$$\tilde{\varphi}(x, \kappa) := f(x) - \kappa\chi(\nabla^2 f(x))$$

where

$$\chi(H) := \sum_{i=1}^{n} \min\{0, \lambda_i(H)\}^2$$

and $\lambda_i(H)$ denotes the i-th eigenvalue of the symmetric matrix H. Hence, χ is the sum of squares of the negative eigenvalues of H.

Let $\sigma : I\!R^n \to I\!R$ be the symmetric (in the sense of permutation) function

$$\sigma(x) := \sum_{i:x_i<0} x_i^2.$$

Denoting $\lambda : \mathbb{S}_n \to \mathbb{R}^n$ the function which assings to every symmetric matrix H the vector of its eigenvalues, i.e. $\lambda(H) = (\lambda_1(H), ..., \lambda_n(H))^T$, we can set the symmetric spectral function corresponding to χ to

$$\sigma(\lambda(H)) = \chi(H) = \sum_{i:\lambda_i(H)<0} \lambda_i(H)^2.$$

By Theorem 1.1. in [1] the function $\chi(H)$ is differentiable, and its derivative is given by

$$\frac{\partial d}{\partial H}\chi(H) = \frac{\partial d}{\partial H}(\sigma \circ \lambda)(H) = U(Diag(\sigma'(\lambda(H))))U^T = U(Diag(2\lambda_-(H)))U^T,$$

where $\lambda_-(H) = (0, ...0, \lambda_{p+1}(H), ..., \lambda_n(H))^T$ (denoting the vector containing only the negative eigenvalues of H, elsewhere 0). We also write shortly

$$H_- = U(Diag(\lambda_-(H)))U^T.$$

In consequence we have the i-th component of the derivative of $\tilde{\varphi}(x,\kappa)$ given by

$$(Df(x))_i - 2\kappa(\nabla^2 f(x))_- \bullet \Delta H_i.$$

where $\Delta H_i := \frac{\partial}{\partial x_i}\nabla^2 f(x)$.

Let $\hat{d}^3$ be the vector with components

$$\hat{d}_i^3 = (\nabla^2 f(x))_- \bullet \Delta H_i$$

and $d^3 = \hat{d}_i^3/\|\hat{d}_i^3\|$. As in Section 2 we construct a search direction Δx having positive scalar products with d^1 and d^3. In contrast to Section 2 we do not choose a nonnegative combination of d^1 and d^3 but restrict the search direction to the space Q.

We therefore obtain a search direction $\Delta x_{dcd} = s$ as a solution of

$$\min_{(s,t)\in\mathbb{R}^{n+1}} \{-t \mid s^T d^3 \geq t, s^T d^1 \geq t, s^T s \leq 1, s \in Q\} \tag{2}$$

maximizing the cosine of the angles $\sphericalangle(\Delta x, d^1)$ and $\sphericalangle(\Delta x, d^3)$ over Q.

Note that this problem is a convex problem, and Slater's condition is satisfied for $(s,t) = 0 \in \mathbb{R}^{n+1}$. Thus, any point satisfying the KKT-conditions is a local (and global) solution. In our numerical tests, we simply used Sedumi ([2]) for solving this subproblem. By exploiting the structure of the subproblem, there certainly exist more efficient solutions.

5 Numerical results

Figure 4 refers to the same example as Figures 1 and 2. The dark (red) shaded area in Figure 4 is the set of starting points from which the curvature descent method converges to the local minimizer $(-1,-2)^T$ while missing the points with lower objective value along the negative x_2-axis. Comparing this plot with Figure 2 seems to indicate that the curvature descent approach is inferior to $igd(0.6)$. More difficult test problems below, however, indicate otherwise.

We point out that the *dcd*-method does not make sense for two-dimensional examples, and it is therefore not plotted here.

Next, we compared the four approaches on test problems derived from Sensor Network Localization Problems with distance measurements (as described for example in [3]).

These nonconvex optimization problems are derived from a satisfyability problem: The data consits of k so called anchor points $a_{(1)},...,a_{(k)} \in \mathbb{R}^2$ and distances d_{ij} and $d_{pq} \in \mathbb{R}$, for $(i,j) \in \mathcal{J}_1 \subset \{1,\ldots,n\} \times \{1,\ldots,n\}$ and $(p,q) \in \mathcal{J}_2 \subset \{1,\ldots,k\} \times \{1,\ldots,n\}$. The problem is to find n vectors (sensors) $x_{(1)},...,x_{(n)} \in \mathbb{R}^2$ such that

$$\begin{aligned} \|x_{(i)} - x_{(j)}\|_2 &= d_{ij} \quad \text{for } (i,j) \in \mathcal{J}_1 \\ \|a_{(p)} - x_{(q)}\|_2 &= d_{pq} \quad \text{for } (p,q) \in \mathcal{J}_2. \end{aligned}$$

Note that typically the distance informations d_{ij},d_{pq} are given only for sensors/anchors within a certain radio range of each other – this radio range defines the sets $\mathcal{J}_1$ and $\mathcal{J}_2$. This problem can be modelled with a semidefinite program with an additional rank condition (rank=2 in [3]). Ignoring the rank

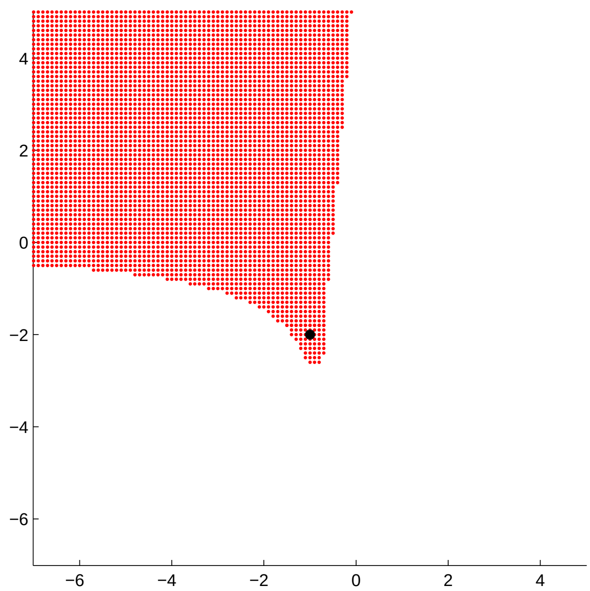

Figure 4: Area indicating starting points from which *cd* convergences to the (poor) local minimizer of f defined in (1).

condition leads to an SDP problem which can be solved by common interior point algorithms. Of course the solution of the SDP problem may not satisfy the rank condition, and is therefore just an approximation to the solution of the sensor location problem. For problems with exact distance information the SDP solution yields a very good approximation to the unknow vectors $x_{(i)}$ – in fact the exact solutions can be retrieved when the network is uniquely localizable, see [4]. But when measurement errors are taken into account, this is generally not true anymore.

For our test problems we randomly generated sensors and anchors in $[-0.5, 0.5]^2$. We then computed the distance information according to a given radio range and added a measurement error in the following way:

$$\tilde{d}_{ij} = d_{ij}(1 + rand(1) * nf)$$
$$\tilde{d}_{pq} = d_{pq}(1 + rand(1) * nf)$$

where nf denotes the noise factor (in most problems $nf = 0.1$, which represents up to 10% measurement error).

We then treated the sensors as unknown, solved the resulting SDP problem with Sedumi and derived from the resulting matrix an initial approximation $\hat{x}$ to the $2n$ dimensional vector of unknown sensors $x_{(1)}, ..., x_{(n)}$. Using the vector $\hat{x}$ as a starting point we search for a solution of the nonconvex approximation problem

$$\min_{x\in\mathbb{R}^{2n}} \sum_{(i,j)} (\tilde{d}_{ij}^2 - \|x_{(i)} - x_{(j)}\|_2^2)^2 + \sum_{(k,l)} (\tilde{d}_{kl}^2 - \|a_{(k)} - x_{(l)}\|_2^2)^2. \tag{3}$$

This problem has many local minimizers. Below, we present computational results comparing the steepest descent method ($\Delta x_{igd(1)}$), with the other approaches: $\Delta x_{igd(\rho)}$ ($\rho = 0.6$), Δx_{cd} and Δx_{dcd}. We used a linesearch with short steps and compared the quality of the solution to the rounding technique presented in [5] (abbreviated here as SDP2). The table lists the "normalized final accuracy" obtained for each of the five approaches where the "normalized final accuracy" is computed as follows: First, the data of the exact problem is used as a starting point for the steepest descent method for minimizing the sum of squares of the (perturbed) residuals (3). The result yields a (typically very good) upper bound for the global minimizer of (3). The normalized final accuracy then lists how far the final accuracy deviates from this upper bound, negative values indicating that a solution was found that is better than the upper bound. (To faciliate the presentation all the final accuracies were multiplied by 100.)

All approaches were stopped when a local minimizer was reached; the final error therefore does not depend on the local behavior of the method but on its global performance.

For the search direction Δx_{dcd} we used the fact that the Hessian is sparse; the computation of the third derivative was very cheap in this case. The Δx_{dcd}-step was changed to Δx_{cd} when the optimal value t in (2) was less than 0.1.

The following table shows the results for 20 random problems with n unknown sensors, k anchor points, radio range rr, and noise factor nf.

n, k, rr, nf	SDP2	Δx_{cd}	Δx_{dcd}	$\Delta x_{igd(1)}$	$\Delta x_{igd(0.6)}$
50, 5, 0.3, 0.15	0.9473	0.9417	**0.7230**	1.733235	**0.7230**
20, 3, 0.35, 0.1	0.1031	**-1.35e-05**	0.0585	**-1.35e-05**	**-1.35e-05**
60, 6, 0.3, 0.1	**-1.08e-07**	**-1.08e-07**	0.4051	**-1.08e-07**	**-1.08e-07**
120, 10, 0.3, 0.1	**-4.10e-05**	**-4.10e-05**	**-4.10e-05**	**-4.10e-05**	**-4.10e-05**
45, 4, 0.35, 0.1	**-1.59e-05**	**-1.59e-05**	**-1.59e-05**	**-1.59e-05**	**-1.59e-05**
45, 5, 0.4, 0.15	**-1.13e-07**	**-1.13e-07**	**-1.13e-07**	**-1.13e-07**	**-1.13e-07**
50, 5, 0.25, 0.15	1.1009	**0.1575**	**0.1575**	**0.1575**	**0.1575**
80, 5, 0.25, 0.1	0.0374	**0.0059**	0.0075	0.0298	**0.0059**
30, 3, 0.25, 0.1	0.0964	**0.0789**	0.1618	0.0876	0.0815
50, 5, 0.3, 0.1	**-6.74e-06**	**-6.74e-06**	**-6.74e-06**	**-6.74e-06**	**-6.74e-06**
50, 3, 0.2, 0.1	0.0663	**0.0372**	0.0502	**0.0372**	**0.0372**
30, 3, 0.3, 0.1	**0.0140**	0.0848	0.0399	0.0848	0.0848
50, 5, 0.3, 0.15	0.83977	**- 0,0018**	0.1709	0.1709	0.1709
30, 3, 0.3, 0.15	**0.4813**	**0.4813**	**0.4813**	**0.4813**	**0.4813**
60, 6, 0.2, 0.15	0.0075	**-0.0029**	0.0220	0.0003	**-0.0029**
50, 5, 0.3, 0.15	0.1291	**-2.99e-04**	**-2.99e-04**	**-2.99e-04**	**-2.99e-04**
30, 3, 0.25, 0.1	0.1066	**0.0647**	0.0907	**0.0647**	**0.0647**
45, 5, 0.3, 0.1	0.0598	0.0598	**0.0563**	0.0598	0.0598
80, 8, 0.25, 0.1	**0.0843**	**0.0843**	0.3637	**0.0843**	**0.0843**
120, 10, 0.1, 0.2	0.0045	**0.0021**	0.0408	0.0026	**0.0021**

This table is representative for most of the problems we tested with other combinations of noise factor, radio range and dimension. As our approaches are just heuristical it is clear, that we cannot expect that *cd* will always yield better (in the sense of a smaller residuum) solutions. Nevertheless we can see a trend to more accurate results.

As an example that in many cases we find not only a solution with a slightly better residuum, but in fact very different local minimizers, we show some graphical interpretation of the first example in the above table. Figures 5 and 6 show the position of the true sensors with a star, the anchors with a square.

Sensors within radio range of each other are connected by a dashed line. The true solutions are marked by a star, the computed solutions by a circle that is connected to the associated star by a blue line.

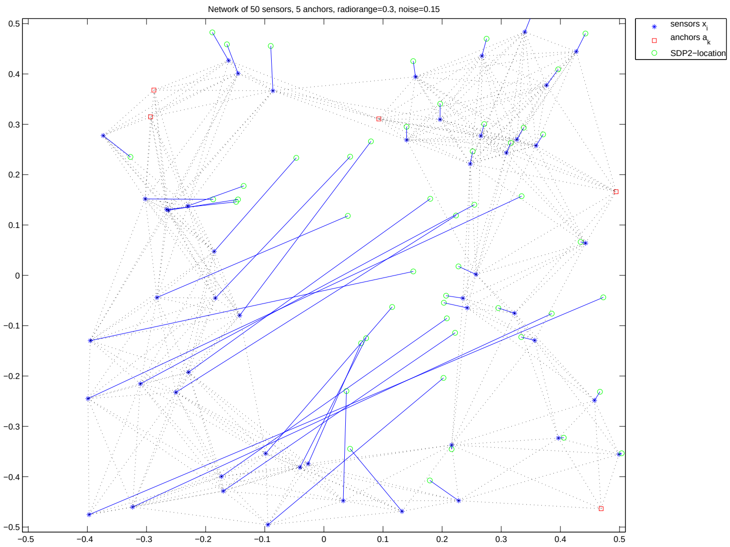

Figure 5: SDP2 approach.

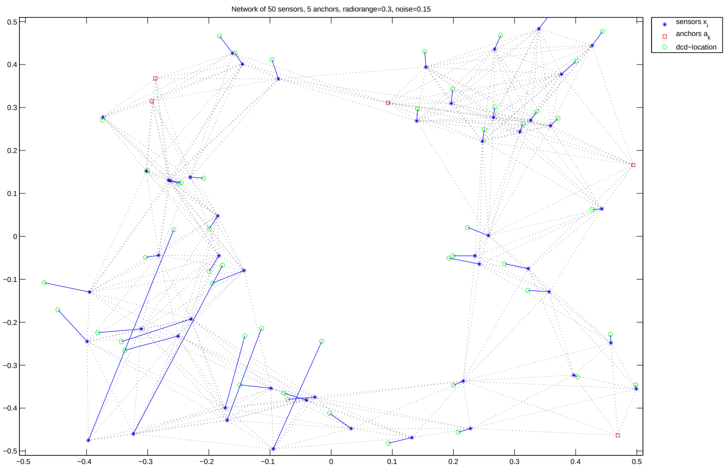

Figure 6: Decreasing Curvature Descent Δx_{dcd}.

6 Conclusion

This paper identifies a very cheap heuristic *idg* for avoiding "poor"local minimizers. A more expensive and often more effective approach is based on *cd*. In various modifications the *cd*-method is widely used for unconstrained minimization. Here, we identify parameters that tend to generate good local solution for problems derived from sensor location. Finally, an approach aiming at increasing the negative curvature proved to be efficient for some of the examples.

Acknowledgements

We would like to thank Yinyu Ye and Zizhuo Wang for kindly sending us their program codes.

References

[1] Lewis, A.S.: Derivatives of Spectral Functions, *Mathematics of Operations Research*, Vol. 21, No. 3, pp. 576-588 (1996).

[2] Sturm, J.F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones, *Optimization Methods and Software*, 11-12, No. 1-4, pp. 625-653 (1999).

[3] Biswas, P.; Ye, Y.: Semidefinite Programming for Ad Hoc Wireless Sensor Network Localization, *Proceedings 3rd IPSN*, pp. 4-54 (2004).

[4] So, A. M.-C.; Ye, Y.: Theory of Semidefinite Programming for Sensor Network Localization, *Mathematical Programming* 109, No. 2-3 (B), pp. 367-384 (2007).

[5] Liang, T.C.; Wang, T.C.; Ye, Y.: A Gradient Search Method to Round the Semidefinite Programming Relaxation Solution for Ad Hoc Wireless Sensor Network Locations, Technical Report, Dept. Management Sci. Eng,, Stanford University (2004).