REPORTS IN INFORMATICS

ISSN 0333-3590

Convergence Basins for Some Derivative-Free Optimization Methods on Problems with Saddle Points

Lennart Frimannslund and Trond Steihaug

REPORT NO 394 **June 2010**

Department of Informatics

UNIVERSITY OF BERGEN

Bergen, Norway

This report has URL
http://www.ii.uib.no/publikasjoner/texrap/pdf/2010-394.pdf

Reports in Informatics from Department of Informatics, University of Bergen, Norway, is available at http://www.ii.uib.no/publikasjoner/texrap/.

Requests for paper copies of this report can be sent to:

Department of Informatics, University of Bergen, Høyteknologisenteret,
P.O. Box 7800, N-5020 Bergen, Norway

Convergence Basins for Some Derivative-Free Optimization Methods on Problems with Saddle Points

Lennart Frimannslund* Trond Steihaug†

June 9, 2010

Abstract

We give a detailed description of some of the numerical results included in the paper *"A subclass of Generating Set Search with convergence to second-order stationary points"*, by Mark A. Abramson, Lennart Frimannslund and Trond Steihaug. Specifically, we study basins of attraction for two functions with saddle points for five different derivative-free optimization methods.

1 Introduction

The purpose of this progress report is to report on numerical experiments on two unconstrained optimization problems where methods risk terminating at a saddle point, avoiding a nearby strict local minimizer. For visualization purposes we have chosen problems with two unknowns.

The first problem is a modification of a problem suggested by Wolfe [10]. In its unmodified form this problem has been used to show that gradient based methods tend to converge to a saddle point. The modification will make the function bounded below and introduce a local minimizer but not change the region where gradient based methods converge to the saddle-point. The second example is a modification of a function presented in [1], which has a very narrow cone of negative curvature. Again the modification will make the function bounded below and introduce local minimizers.

A generating set search method was introduced in [2] which was shown to converge to second-order stationary points, and should therefore not experience problems on the test functions. It is compared with two methods which do not have the same theoretical convergence properties.

2 The methods

The three methods primarily used in testing, are GSS-CI, NEWUOA and NMSMAX. We will briefly discuss two additional methods, MDSMAX and fminsearch, in Section 5.

2.1 GSS-CI

This is the method presented in [2], and is based on the method of [3]. It can be thought of as compass search (see e.g. [5]) with adaptive search directions. Through finite difference computations using the function values at previously sampled points, average curvature information is gathered in a Hessian-like

*Department of Mathematics, University of Bergen, Norway. E-mail: lennart@ii.uib.no.
†Department of Informatics, University of Bergen, Norway. E-mail: trond@ii.uib.no.

matrix, and the eigenvectors of this matrix are then used as the search directions. The process is then repeated, so the search directions can change many times. The method also gathers average slope information, and consequently can perform Newton-like steps at regular intervals.

The initial search directions are chosen to be the positive and negative coordinate vectors. Each pair of search directions (e.g. $\pm q_i$, where q_i is a search direction) has a step length δ_i associated with it. In our experiments these are initially set to the same value, $0.2\|x_0\|_1$, but they will be increased or decreased individually depending on the success or failure of the search along the corresponding pairs of search directions. A search is deemed successful it it produces sufficient decrease, that is, if

$$f(x + \delta_i q_i) < f(x) - \rho(\delta_i),$$

where $\rho : \mathbb{R} \mapsto \mathbb{R}$ is nondecreasing function satisfying a few technical requirements, outlined in [5]. In our implementation we use

$$\rho(\delta) = 10^{-4}\delta^2.$$

The termination criterion is that the product of all the step lengths should be less than or equal to a tolerance. In our experiments this is

$$\prod_{i=1}^{n} \delta_i \leq \left(10^{-4}\|x_0\|_1\right)^n.$$

2.2 NEWUOA

NEWUOA [8] is an interpolation method, where the number of interpolation points can be determined by the user. The remaining degrees of freedom are taken up by minimizing the Frobenius norm of the difference between one Hessian approximation and the next.

An initial vector $x_0 \in \mathbb{R}^n$, the number m of interpolation points, and the initial and final values of a trust region radius, namely ρ_{beg} and ρ_{end} must be provided by the user. The number of interpolation points m is a fixed integer from the interval $n + 2 \leq m \leq \frac{1}{2}(n + 1)(n + 2)$. It is recommended to use $m = 2n+1$ for efficiency. The initial interpolation points $x_i, i = 0, 1, 2, \ldots, m$, have the property that $\|x_i - x_0\|_2 = \rho_{\text{beg}}, i = 1, 2, \ldots, m$, unless $m > 2n + 1$, in which case this distance is $\sqrt{2}\rho_{\text{beg}}$. The termination criterion is related to the radius ρ_{end}.

2.3 NMSMAX

NMSMAX [4] is an implementation by Nicholas J. Higham of the classical Nelder-Mead simplex method [7]. The user can choose whether the initial simplex is right-angled or regular (with sides of equal length). The initial simplex size is not input by the user, but taken to be the order of $\max(\|x_0\|_\infty, 1)$. The method terminates when either the maximum number of function evaluations is reached, or when the relative size of the simplex, is below a certain threshold. That is,

$$\frac{1}{\max(1, \|v_0\|_1)} \max_{1 \leq i \leq n} \|v_i - v_0\|_1 \leq \text{tol}.$$

Here v_0 and $v_i,\ i = 1, \ldots, n$ are the vertices of the simplex. In our experiments we use $\text{tol} = 10^{-6}\|x_0\|_1$.

3 The functions

The two functions are in two variables, are twice continuously differentiable and bounded below.

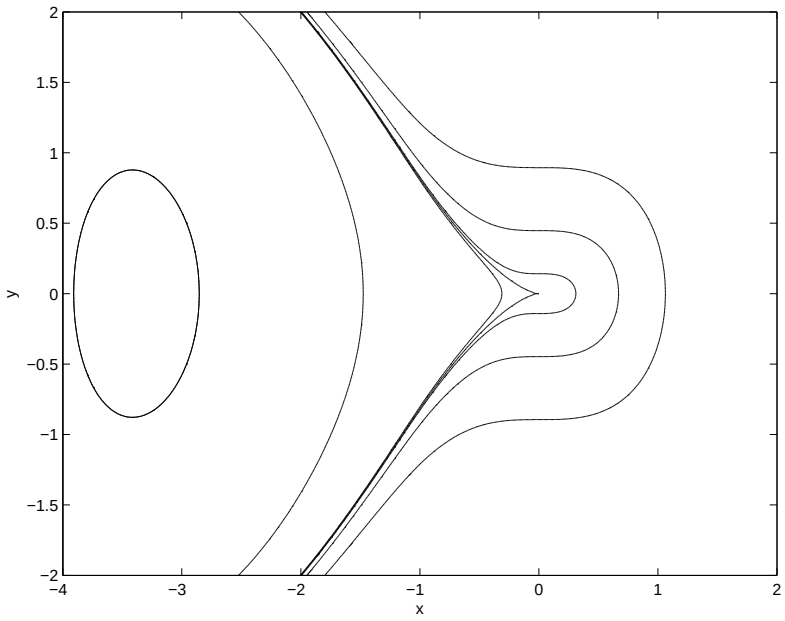

Figure 1: Level curves for the function (2).

3.1 Function I – A Narrow Positive Cone

The function (1) is a modification of a test function in [1]:

$$f(x,y) = (9x-y)(11x-y) + \frac{x^4}{2}. \tag{1}$$

It has a saddle point at the origin, and two local minimizers at $(x,y) = \pm(1,10)$. Level curves for this function can be seen in Figure 6.

3.2 Function II – Modified Wolfe Function

The second function (2) is a modified version of a test function due to Wolfe [10]:

$$f(x,y) = \frac{x^3}{3} + \frac{y^2}{2} - \frac{2}{3}(\min[x,-1]+1)^3, \tag{2}$$

It has a saddle point at the origin and a minimum at $(x,y) = (-2-\sqrt{2},0)$. Level curves for this function are shown in Figure 1. The original function (not bounded below) is given by $f(x,y) = \frac{x^3}{3} + \frac{y^2}{2}$.

4 Numerical experience

Region of Convergence The region of convergence of a stationary point is the set of starting points for which a given method terminates close to the stationary point. In addition to the input parameters for the methods we need to specify the tolerance (or distance between) the terminating point and the stationary point. A globally convergent method on a sufficiently smooth function is characterized by for all starting points, the method will for any $\varepsilon > 0$ generate an iterate x_k so that $\|\nabla f(x_k)\| \leq \varepsilon$. However, the stopping criteria of the implementation may be based on changes in the function values or on the difference between two iterates. Even the case $\|\nabla f(x_k)\| \leq \varepsilon$ will in general not guarantee that the distance between the stationary point and x_k is small. We can thus expect that even if the methods terminate successfully,

the distance to a stationary point will not be smaller than the tolerance for some starting points. For simplicity we say that a method terminates at a stationary point when it terminates at a point that satisfies the tolerance.

4.1 Function I

For this function we generate starting points in the fourth quadrant. The minimizers of the function are in the first and third quadrants, so we expect the methods to terminate successfully at the minimizers if started in these quadrants. This is confirmed in the preliminary numerical testing. Furthermore, because of the symmetry of the function we can choose either the second or fourth quadrants, at least for GSS-CI and NEWUOA.

GSS-CI We discretize the area $[-8, 0] \times [0, 10]$, into a 201×201 grid of points, and start the method with initial step length $0.2\|x_0\|_1$ for all directions, and the termination criterion is that the product of all the step lengths should be less than or equal to $(10^{-4}\|x_0\|_1)^n$. (If $x_0 = 0$ then nonzero values are used.) The results are given in Figure 2. In the figure, a blue color means

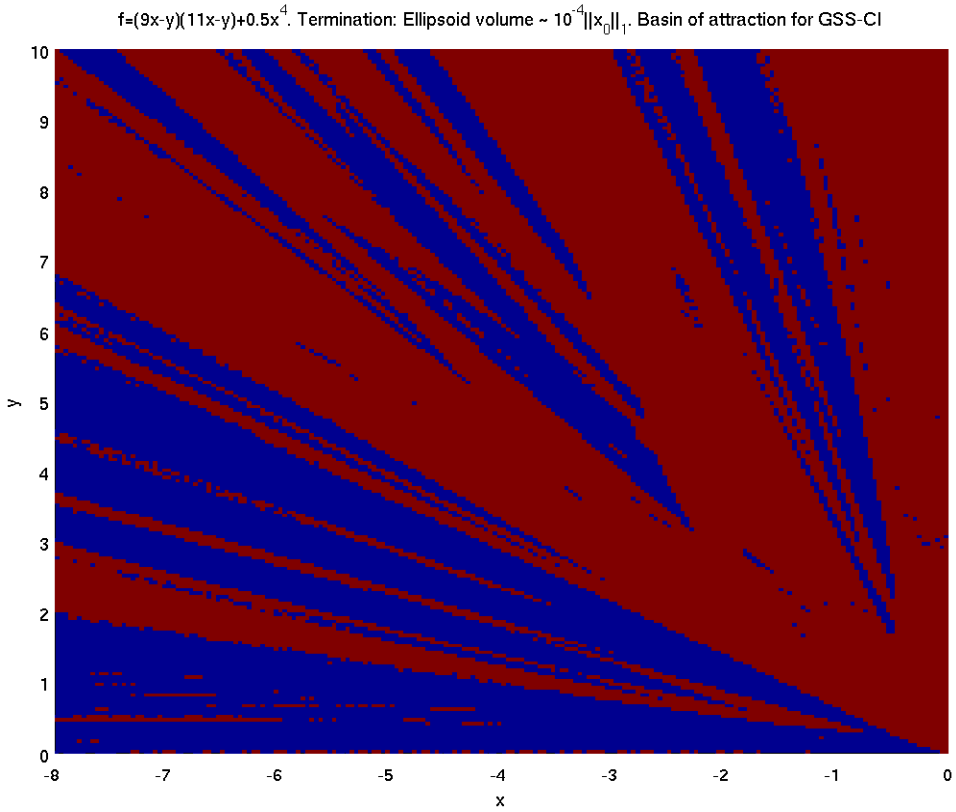

Figure 2: GSS-CI on the function $f = (9x_1 - x_2)(11x_1 - x_2) + \frac{1}{2}x_1^4$. The initial step length is $0.2\|x_0\|_1$ for all directions, and the termination criterion that the volume of the ellipsoid defined by the scaled search directions should be proportional to $10^{-4}\|x_0\|_1$, that is, $\prod_i \delta_i \leq [10^{-4}\|x_0\|_1]^n$. (The discretization is 201×201.)

that the method terminated close to $(x, y) = (1, 10)$ for the corresponding starting point, red color means termination close to $(x, y) = (-1, -10)$. As

one can see, the method does not terminate close to the saddle point for any of these starting points. When starting at the origin, setting a nonzero step length results in convergence to a minimizer.

NEWUOA We generate starting points on a 1001×1001 discretization of the region $[-8, 0] \times [0, 10]$, and run NEWUOA with parameters $\rho_{\text{beg}} = 0.2\|x_0\|_1$, and $\rho_{\text{end}} = 10^{-5}\|x_0\|_1$. (Once again, if $x_0 = 0$ then nonzero values are used.) The results are visualized in Figure 3. As before, blue color means

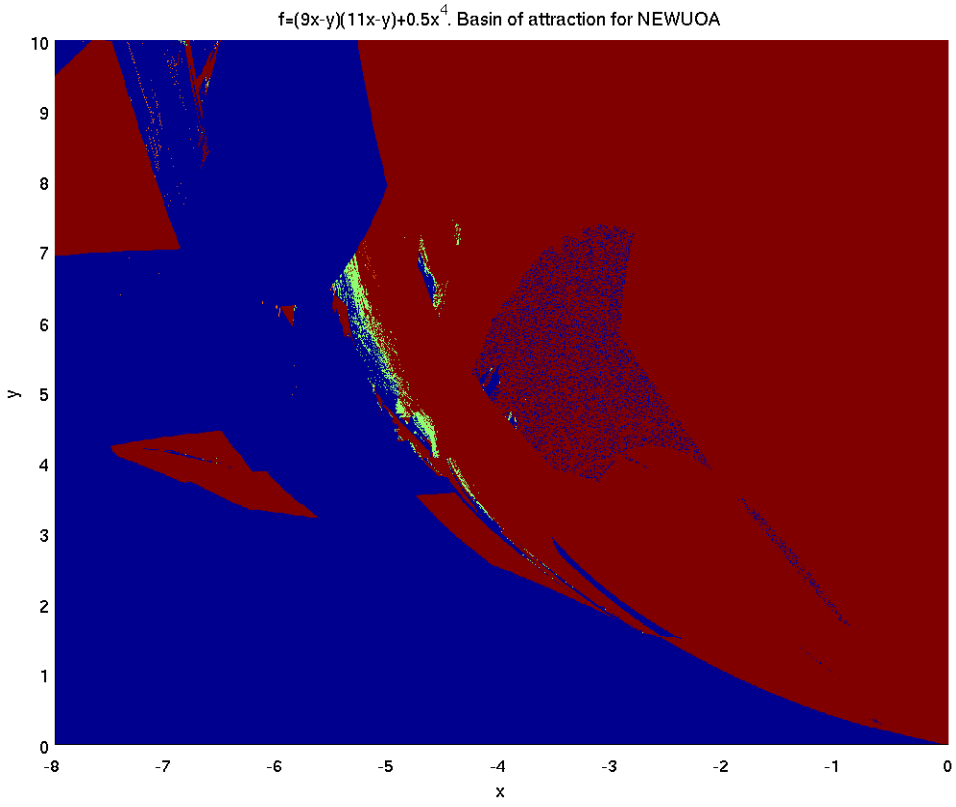

Figure 3: Plot of basins of attraction for NEWUOA on $f = (9x_1 - x_2)(11x_1 - x_2) + \frac{1}{2}x_1^4$, with $\rho_{\text{beg}} = 0.2\|x_0\|_1$ and $\rho_{\text{end}} = 10^{-5}\|x_0\|_1$. (The discretization is 1001×1001.)

that the method terminated close to $(x, y) = (1, 10)$ for the corresponding starting point, red color means termination close to $(x, y) = (-1, -10)$. In addition, green means termination close to the origin, and orange means none of the above. As one can see, the method does terminate close to the saddle point for some starting points, and these points make up a small region on the border between the basins of attraction of $(x, y) = (1, 10)$ and $(x, y) = (-1, -10)$.

To check if the basins of attraction are sensitive to the termination criterion we repeat the experiment, but this time with $\rho_{\text{end}} = 10^{-6}\|x_0\|_1$. The results are given in Figure 4. As we can see in this figure, the starting points for which the method terminates at the saddle point are still wedged between the red and blue regions, but the green region is now much smaller.

Similarly, we test what happens with a looser convergence criterion, namely $\rho_{\text{end}} = 10^{-4}\|x_0\|_1$. The results are in Figure 5. For this convergence criterion

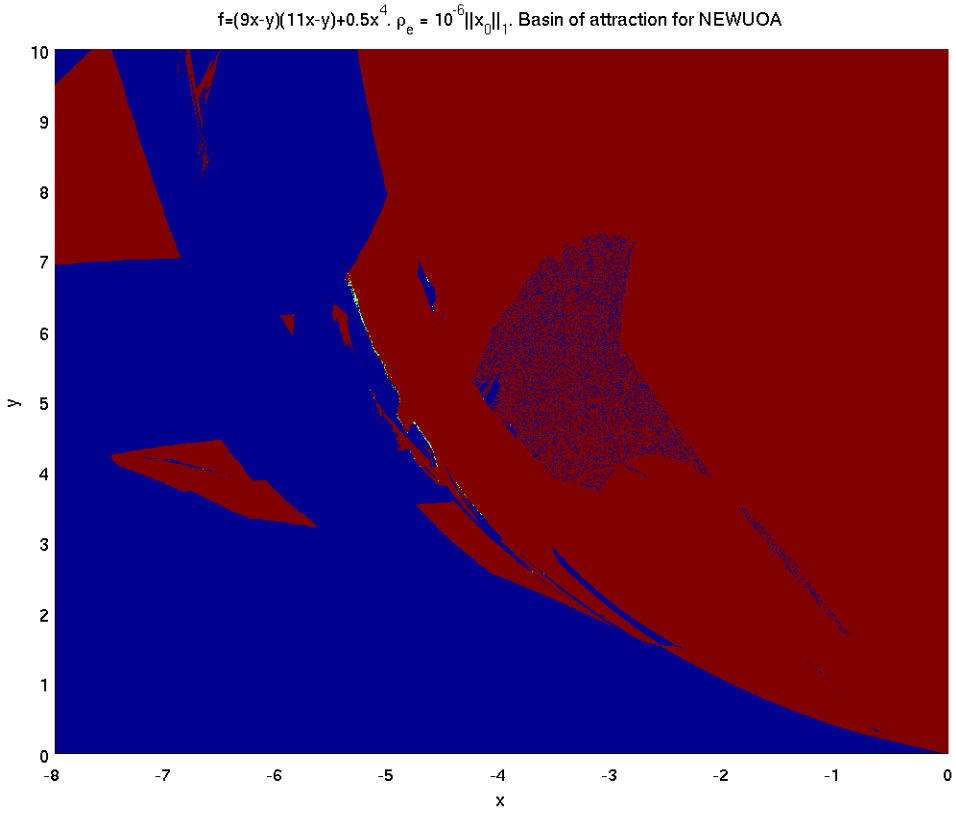

Figure 4: Decreasing ρ_{end} in NEWUOA to $10^{-6}\|x_0\|_1$ will basically not change the region of convergence for the local minimizers, but the region of convergence to the saddle-point gets smaller, squeezed between the regions of convergence to the local minimizers. (The discretization is 1001×1001.)

the green region is much larger, and there are also large orange regions, which correspond to termination no closer to any of the stationary points than 0.2.

NMSMAX For NMSMAX, we discretize the region $[-10, 10] \times [10, 10]$ into a 201×201 grid. We choose a right-angled initial simplex. The size of the initial simplex is not determined by the user, but we set the termination criterion to be a simplex size of $10^{-6}\|x_0\|_1$. The results, as well as level curves of the function are given in Figure 6. As one can see, for about half the fourth quadrant the method terminates at the saddle point, even though the convergence criterion is quite strict. In addition, termination at the saddle point occurs for points close to the negative y-axis , and along the line $y = -x$.

It is also interesting to note that in this case, the behavior in quadrants two and four are *not* the same.

4.2 Function II

For this function we discretize the region $[-4, 2] \times [-2, 2]$ into a 601×401 grid.

GSS-CI Using the same parameter settings as for function I, GSS-CI once again terminates at the (single) minimizer, so an attraction basin plot would

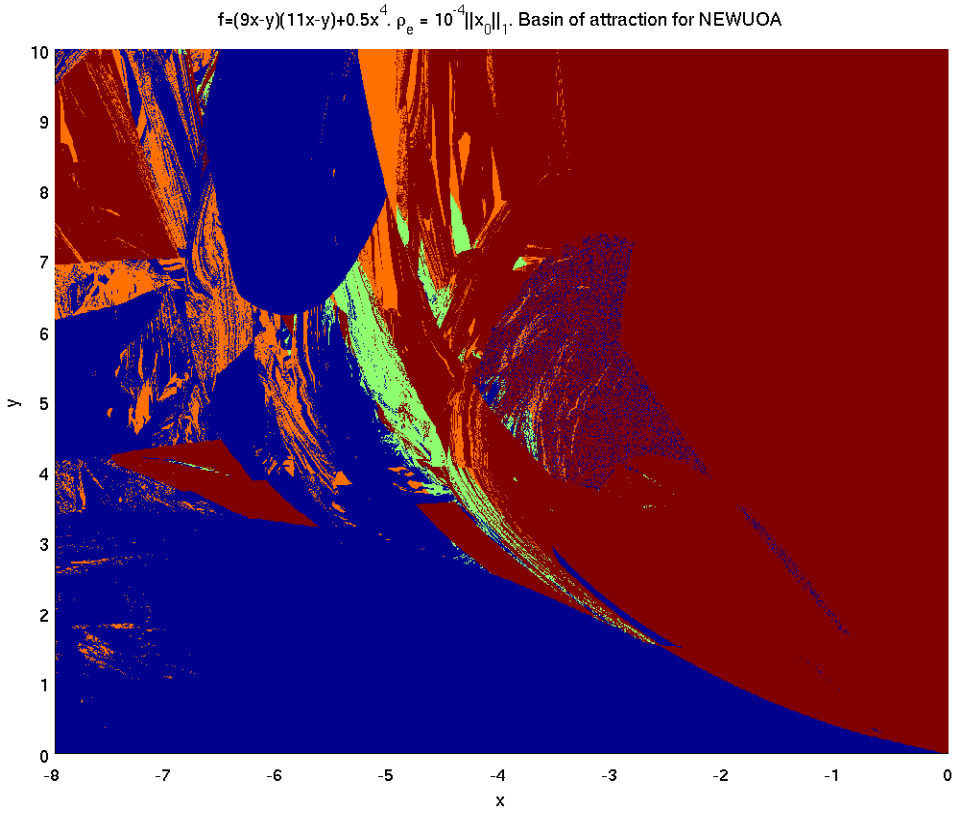

Figure 5: Increasing ρ_{end} in NEWUOA to $10^{-4}\|x_0\|_1$ will force many starting points not to be accepted as close to a stationary point. The regions are basically the same, but the region of convergence for the saddle-point is larger. (The discretization is 1001×1001.)

simply be the region filled with one color. (When $x_0 = 0$, nonzero step lengths are used, and the method then converges to the minimizer.)

NEWUOA For this function we also use the same parameter values as before, namely $\rho_{\text{beg}} = 0.2\|x_0\|_1$ and $\rho_{\text{end}} = 10^{-6}\|x_0\|_1$. The results are in Figure 7. As one can see, there is a relatively large collection of points in the first and second quadrants, for which the method terminates at the saddle point at the origin.

To see if the cause of this behavior was the number of interpolation points ($2n + 1$ in this case), we also tried a full quadratic model, by using six interpolation points. The results for this case are in Figure 8. As one can see, the black region now has a different shape, but is located approximately in the same position, and is of similar size.

NMSMAX For this function, NMSMAX terminates at the saddle point for a few starting points on the y-axis only.

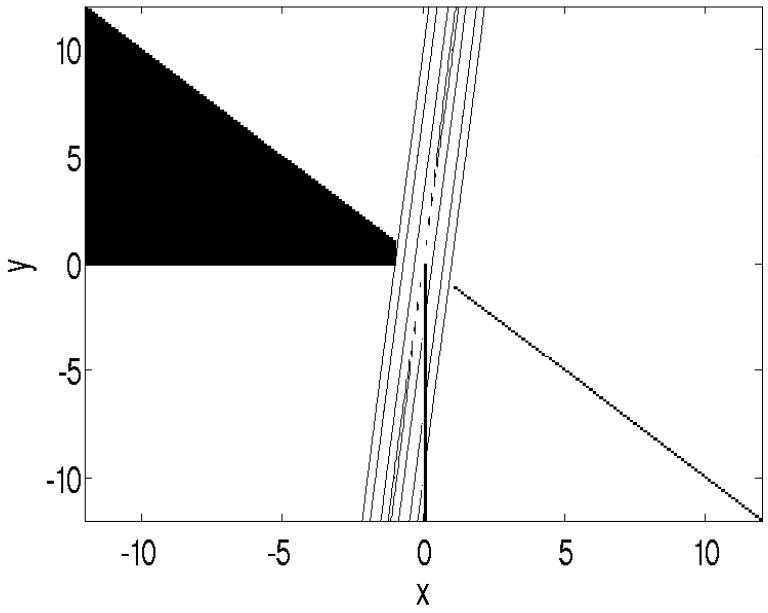

Figure 6: Level curves of the function (1), and starting points for which NMSMAX terminates at the saddle point at the origin, marked in black.

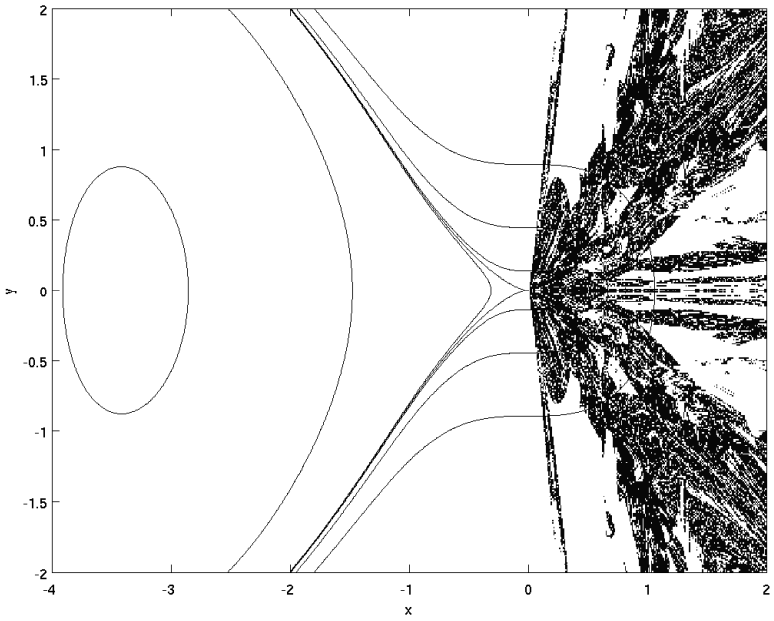

Figure 7: Level curves for the function (2) as well as starting points for which NEWUOA terminates at the saddle point at the origin, marked in black.

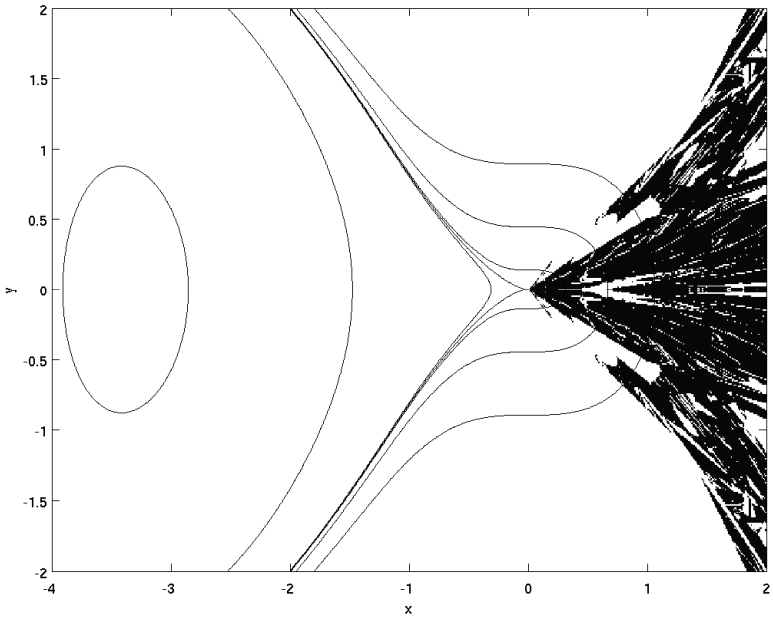

Figure 8: Level curves for the function (2) as well as starting points for which NEWUOA terminates at the saddle point at the origin, marked in black. Number of interpolation points $m = 6$.

5 The methods MDSMAX and fminsearch

We also conducted a brief test of the methods MDSMAX, which is an implementation by Nicholas J. Higham of the multidirectional search method due to Virginia Torczon [9], and fminsearch [6], which is the Matlab implementation of the Nelder-Mead method.

MDSMAX The results are reported in Figure 9 and Figure 10. For the function (2) termination close to the saddle points rarely occurs, and when it does the corresponding starting points lie along straight lines, one at the upper right corner of Figure 10, and one on the y-axis close to the bottom of the figure. However, MDSMAX has serious problems with stagnation on the function (1), as can be seen in Figure 9.

fminsearch This method has few problems on these functions, except when the starting points lie on one of the axes. For the function (1), 208 of the 40401 starting points result in termination close to the saddle point, 201 of these 208 points being on the x-axis. For the function (2), 890 of the 241001 starting points result in termination close to the saddle point, all of these on or immediately next to the y-axis. These results were obtained using the standard convergence tolerances. Tightening the convergence criteria gives an even more favourable result.

6 Discussion

The simplex method [7] is one of the most used derivative free optimization methods. In this note we have used the implementation NMSMAX. The methods NEWUOA and NMSMAX may terminate close to the saddle point while GSS-CI will not converge to the saddle point for these two examples.

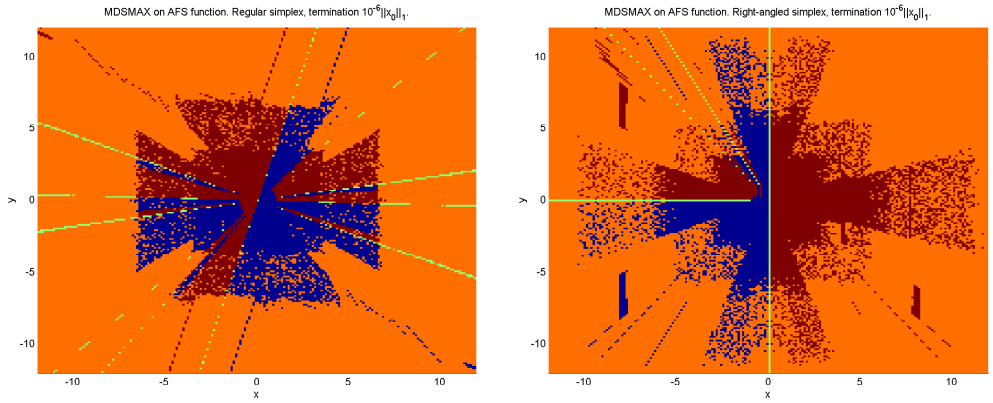

Figure 9: MDSMAX on the function $f = (9x - y)(11x - y) + \frac{x^4}{2}$. Red means termination close to $(x, y) = (1, 10)$, blue means termination close to $(x, y) = (-1, -10)$, green termination close to the saddle point at the origin, and orange means stagnation. Regular simplex on the left, and right-angled simplex on the right.

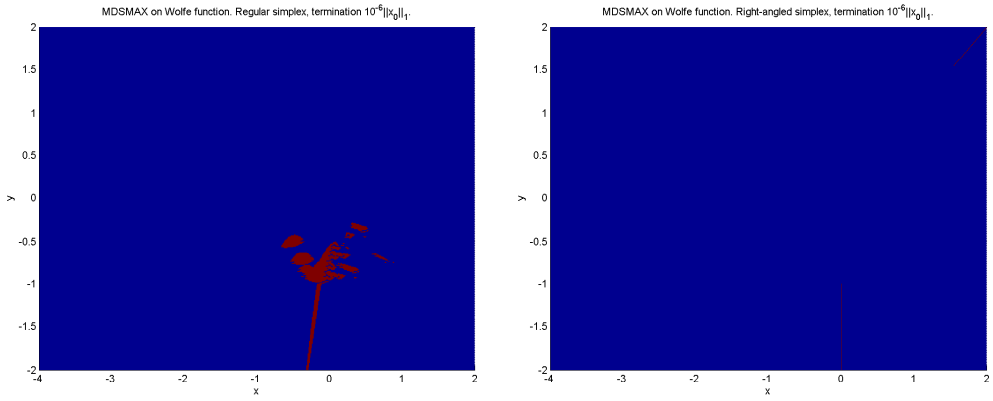

Figure 10: MDSMAX on the function $f = \frac{x^3}{3} + \frac{y^2}{2} - \frac{2}{3}(\min[x, -1] + 1)^3$. Blue means termination close to the minimum at $(x, y) = (-2 - \sqrt{2}, 0)$, red means termination close to the saddle point at the origin. Regular simplex on the left, right-angled simpled on the right.

This supports the theoretical convergence properties of GSS-CI. Similar testing of other implementations of the simplex method reveals that the method may indeed terminate close to saddle points. The regions of convergence are dependent on the the input parameters.

Acknowledgements

The authors would like to thank Mike Powell and Nicholas J. Higham for helpful comments on the experiments and on an earlier version of this manuscript.

References

[1] M. A. Abramson. Second-order behavior of pattern search. *SIAM J. Optim.*, 16(2):315–330, 2005.

[2] M. A. Abramson, L. Frimannslund, and T. Steihaug. A subclass of generating set search with convergence to second-order stationary points. To be submitted, 2010.

[3] L. Frimannslund and T. Steihaug. A generating set search method using curvature information. *Comput. Optim. Appl.*, 38(1):105–121, 2007.

[4] N. J. Higham. The Matrix Computation Toolbox. `http://www.ma.man.ac.uk/~higham/mctoolbox`.

[5] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45:385–482, 2003.

[6] The MathWorks, Inc., Natick, Massachussets, USA. *Optimization Toolbox User's Guide*, 2010.

[7] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7:308–313, 1965.

[8] M. J. D. Powell. *Large-Scale nonlinear optimization*, volume 83 of *Nonconvex Optimization and its applications*, chapter The NEWUOA software for unconstrained optimization without derivatives, pages 255–297. Springer US, 2006.

[9] V. Torczon. *Multi-Directional Search: A Direct Search Algorithm for Parallel Machines*. PhD thesis, Department of Mathematical Sciences, Rice University, Houston, Texas, 1989. Available as Tech. Rep. 90-07, Department of Computational and Applied Mathematics, Rice University, Houston, Texas 77005-1892.

[10] P. Wolfe. Convergence conditions for ascent methods. II: Some corrections. *SIAM Review*, 13(2):185–188, 1971.

Appendix: Making attraction basin plots

This appendix is used in the course Nonlinear optimization at the Department of Informatics at the University of Bergen, Norway.

In order to make attraction basin plots we need a function and an optimization routine. Attraction basin is also called region of convergence. We say that the region of convergence for a particular stationary point x^* is all starting points x_0 so that the methods terminate close to the stationary point x^*. When a method terminates, it can either be close to some stationary point or too far away. Let x_k be the final iterate (with the starting point x_0) we have a successful starting point if $\|x^* - x_k\| \leq \varepsilon_p$. This is a proximity measure with parameter ε_p.

Consider the function

$$f(x_1, x_2) = (9x_1 - x_2)(11x_1 - x_2) + \frac{1}{2}x_1^4.$$

It has three stationary points, minima at $(x_1, x_2) = \pm(1, 10)$, and a saddle point at the origin.

In Matlab the objective function can be:

```
function [y] = narrowValley(x);
y = (9*x(1) - x(2))*(11*x(1)-x(2)) + 0.5* x(1)^4;
return;
```

As the name suggests, it contains a very narrow valley which can trap an optimization method to terminates in the valley away from the stationary points.

As for the optimization routine, Matlab has a built-in method for unconstrained optimization without derivatives, called fminsearch. One of its calling sequences is:

```
[x_opt,f_opt] = fminsearch(f,x0,options);
```

Here "options" is a Matlab structure which contains the options for fminsearch. It is created with the command optimset. The function is stored in a file with the name narrowValley.m. If we wanted to minimize the function starting at, say, $(x_1, x_2) = (-2, 2)$, we could write:

```
x0 = [-2 2]';
f = @narrowValley;
myOptions = optimset('TolX',1e-6,'TolFun',1e-6);
[x_opt,f_opt] = fminsearch(f,x0,myOptions);
```

Now the best x and corresponding f-value are stored in the two output variables. It it not strictly necessary to define and use the myOptions structure, but the options used here make the convergence criteria more strict than the default settings.

Solving the problem for many starting points With the objective function defined and having selected an optimization method we are ready to gather the data which will form the basis for the attraction basin plot. We will discretize the area of interest by having vectors for the x- and y-values used, and store a diagnostic value based on the proximity to the stationary points in a matrix, A. This is done in the following code:

```
% Fineness of the discretization, larger is finer
grid_fineness = 200;

% Limits for x
xmax = 12;
xmin = -12;
dx = (xmax-xmin) / grid_fineness;
```

```
% Limits for y
ymax = 12;
ymin = -12;
dy = (ymax-ymin) / grid_fineness;

% The discretization itself
x = xmin:dx:xmax;
y = ymax:-dy:ymin;

lx = length(x);
ly = length(y);

% Matrix for storing results
A = zeros(ly,lx);

% Stationary points, to be used in postprocessing.
% These depend on the objective function and must be
% known a priori.
xplus  = [ 1  10]';
xminus = [-1 -10]';
xzero  = [ 0   0]';

% Values to be used for plotting
xplus_plotting_value = 10;
xminus_plotting_value = -10;
xzero_plotting_value = 0;
other_plotting_value = 5;

% The maximum distance we allow a termination point to have to a
% stationary point
proximity = 0.2;


% Run through all our starting points
for (i=1:ly),
  for (j=1:lx),

    % Current starting point
    x0 = [x(j) y(i)]';

    % Set options for solver, may or may not depend on starting point
    myOptions = optimset('TolX',1e-6,'TolFun',1e-6);
    % And off we go!
    [best_x, best_f] = fminsearch(f, x0, myOptions);

    % Our termination point is now in best_x, test for its
    % proximity to the stationary points
    if (norm(best_x-xplus) <= proximity),
      A(i,j) = xplus_plotting_value;
    elseif (norm(best_x - xminus) <= proximity),
      A(i,j) = xminus_plotting_value;
    elseif (norm(best_x - xzero) <= proximity),
      A(i,j) = xzero_plotting_value;
    else,
      A(i,j) = other_plotting_value;
    end;
  end;
```

```
end;
```

Plotting the results With the results now in a the matrix A we can visualise the results using the Matlab command surf. This command can plot both two- and three-dimensional plots, and by default uses OpenGL to do this. This may cause Matlab to crash and cause the user to be logged out of his/her session when doing this on the University of Bergen's system. For this reason, include the first line in the following code:

```
% to prevent OpenGL from crashing Matlab
figure; set(gcf,'renderer','zbuffer');

% good to have on one line, otherwise view(2)
% can be effectively ignored
surf(x,y,A,'EdgeColor','none'); view(2)

xlabel('x'); ylabel('y');
title('Attraction basin plot');
```

It may happen that Matlab seemingly ignores the commands xlabel, ylabel and title. This is because the figure isn't refreshed. This can be solved by running these commands in a script instead of typing them manually. Alternatively, all of these commands can be typed on one line. (This is also why the view command is on the same line as the surf command.)