INTRODUCTION TO FUNCTION OPTIMISATION

JOSE BERNAL

Finding the Minimum of Functions

Department of Electrical Engineering
University of Burgundy

December 2015 – version 1.0

Jose Bernal: *Introduction to Function Optimisation,* Finding the Minimum of Functions, © December 2015

PRESENTED TO:
PhD Yohan Fougerolle

ABSTRACT

Short summary of the contents...

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— ? [?]

ACKNOWLEDGEMENTS

Put your acknowledgements here.

Many thanks to everybody who already sent me a postcard!

Regarding the typography and other help, many thanks go to Marco Kuhlmann, Philipp Lehman, Lothar Schlesier, Jim Young, Lorenzo Pantieri and Enrico Gregorio[1], Jörg Sommer, Joachim Köstler, Daniel Gottschlag, Denis Aydin, Paride Legovini, Steffen Prochnow, Nicolas Repp, Hinrich Harms, Roland Winkler, and the whole LaTeX-community for support, ideas and some great software.

Regarding LyX: The LyX port was intially done by *Nicholas Mariette* in March 2009 and continued by *Ivo Pletikosić* in 2011. Thank you very much for your work and the contributions to the original style.

1 Members of GuIT (Gruppo Italiano Utilizzatori di TeX e LaTeX)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

DRY Don't Repeat Yourself

API Application Programming Interface

UML Unified Modeling Language

Part I

FUNCTION OPTIMISATION

You can put some informational part preamble text here. Illo principalmente su nos. Non message *occidental* anglo-romanic da. Debitas effortio simplificate sia se, auxiliar summarios da que, se avantiate publicationes via. Pan in terra summarios, capital interlingua se que. Al via multo esser specimen, campo responder que da. Le usate medical addresses pro, europa origine sanctificate nos se.

1 INTRODUCTION

1.1 BOLZANO'S THEOREM

The Bolzano's theorem is cornerstone in function optimisation (in particular, root finding). This theorem is graphically presented in Fig. **??**. In a few words, given a function f defined in the interval $[a, b]$, if the sign of $f(a)$ and $f(b)$ changes, then there is a root of f in that interval.

Part II

THIS IS THE SECOND PART

You can put some informational part preamble text here. Illo principalmente su nos. Non message *occidental* anglo-romanic da. Debitas effortio simplificate sia se, auxiliar summarios da que, se avantiate publicationes via. Pan in terra summarios, capital interlingua se que. Al via multo esser specimen, campo responder que da. Le usate medical addresses pro, europa origine sanctificate nos se.

2

OPTIMISATION OF FUNCTIONS IN ONE DIMENSION

2.1 BRUTE FORCE APPROACH

A naive algorithm for 1D minimisation consists in taking samples on a given interval and selecting the minimum among them. This brute force approach presents several drawbacks. First, the sampling frequency should be small enough in order to obtain a value close to the minimum. Depending on the concept of enough, this method may turn into an inefficient algorithm. Second, there is no strategy supporting the sampling method. Thus, it can luckily find the global minimum. Taking the previous facts into account, it is not interesting to analyse and, hence, no further discussion about this method is presented in this work.

2.2 INTERVAL-BASED MINIMUM FINDING

In order to understand the interval-based approach for minimum finding, we recall the approximation of the bisection algorithm for root finding. The algorithm determines the minimum of an unimodal function f on a close interval $[a, b]$ by successively splitting it into two parts and selecting the one fulfilling the bolzanos' theorem (Section 1.1).

The concept can be extended to determine whether there is a minimum on the range or not. If x is a minimum in $[a, b]$, then

$$f(x) \leqslant f(x_i), \forall\, x_i \in [a, b]. \tag{1}$$

Thus, for every pair of points c and d in $[a, b]$, the conditions $f(x) \leqslant f(c)$ and $f(x) \leqslant f(d)$ are fulfilled. Conveniently, we constraint these points such that $c < x < d$ defining a triplet (c, x, d) in which the minimum is enclosed. Therefore, the goal of an interval-based minimum finding algorithm is to reduce the initial triplet (instead of a range in the case the bisection algorithm) iteratively. The sketch of this type of algorithms is presented in Algorithm 1.

Algorithmus 1 : Interval-based minimum finding algorithms

Data : Triplet (a_0, b_0, c_0)
Result : A minimum b_n in the given interval

```
n = 0;
while (a_n, b_n, c_n) is not small enough do
    calculate fourth point x;
    if (a_n, b_n, x) contains minimum then
        set (a_{n+1}, b_{n+1}, c_{n+1}) equal to (a_n, b_n, x);
    else
        set (a_{n+1}, b_{n+1}, c_{n+1}) equal to (b_n, x, c_n);
    set n equal to n + 1
```

The strategy for deciding how to select the x presented in the previous algorithm is addressed differently. It can be based on optimal divisions in terms of proportions such as in the Golden Search method or parabolic interpolation such as in the Brent's method. The two strategies are described in the following sections.

2.2.1 *Golden Search method*

The strategy of the Golden Search method is based on the Golden ratio. Given the triplet (a, b, c), the value of x corresponds to a fraction 0.38197 into the larger of the two intervals ($[a, b]$ or $[b, c]$) measured from the central point of the triplet.

$$x = \begin{cases} b - \phi * (b - a) & \text{if } |b - a| > |c - b|, \\ b + \phi * (c - b) & \text{otherwise} \end{cases} \text{, where } \phi = 0.38187. \quad (2)$$

2.2.2 *Brent's method*

The idea behind the Brent's method to find the minimum of a function passing through the points of the triplet. Since three points are given, a parabola can be determined using Lagrange interpolation. However, instead of using the data directly, each component of the triplet is inverted and inverse quadratic interpolation is used [1]. Thus, the idea is to obtain x such that the function is minimized. The final expression is presented as follows:

$$x = c - \frac{1}{2} \cdot \frac{(c - a)^2[f(c) - f(b)] - (c - b)^2[f(c) - f(a)]}{(c - a)[f(c) - f(b)] - (c - b)[f(c) - f(a)]}. \quad (3)$$

One of the main drawbacks of this method is that $f(a)$, $f(b)$ and $f(c)$ should be different. Otherwise, the denominator would be equal to 0.

2.3 DERIVATIVE-BASED MINIMUM FINDING

2.3.1 *Gauss-Newton's Method*

The Gauss-Newton method is based on the Taylor series representation of a function. Given a function $f(x)$ infinitely differentiable, its expression through Taylor series is expressed as:

$$f(x+\epsilon) = \sum_{n=0}^{\infty} \frac{\epsilon^n}{n!} \cdot f^{(n)}(x). \tag{4}$$

The algorithm assumes that a quadratic function can be approximated in the region and, thus, the expression is reduced to two terms:

$$f(x+\epsilon) \approx f(x) + \epsilon \cdot f'(x) + \frac{\epsilon^2}{2} \cdot f''(x). \tag{5}$$

The characteristic of a minimum or maximum of a function is that the derivative is equal to zero. The derivative can be also expressed in terms of Taylor series by deriving Eq. 5:

$$\frac{d}{d\epsilon} f(x+\epsilon) \approx \frac{d}{d\epsilon}[f(x) + \epsilon \cdot f'(x) + \frac{\epsilon^2}{2} \cdot f''(x)] \tag{6}$$

$$0 = f'(x) + \epsilon \cdot f''(x) \tag{7}$$

$$\epsilon = -\frac{f'(x)}{f''(x)}. \tag{8}$$

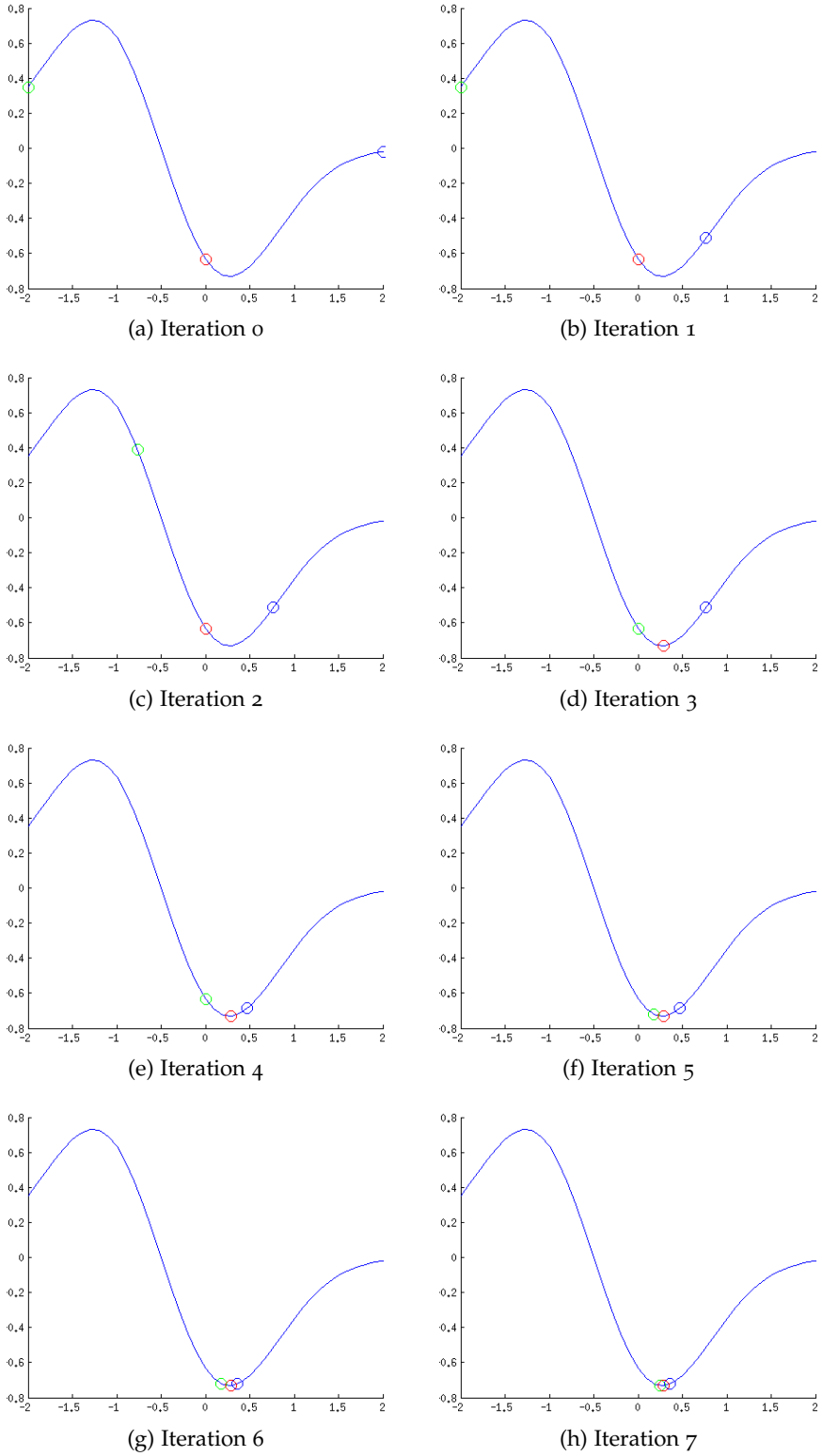
The selection of the of the next point to analyse $x_n + 1$ is presented in the following expression:

$$x_{n+1} = x_n + \epsilon \tag{9}$$

$$x_{n+1} = x_n - \frac{f'(x)}{f''(x)}. \tag{10}$$

Although the method itself is simple, there are some points that should be taken into account:

- The method requires that the function is differentiable at least twice such that the interpolation of order 2 can be performed.
- The second derivative should not be close to 0 on a certain point. Otherwise, the value of ϵ will tend to ∞. This situation generates the well-known over-shooting problem. This is, in fact, one of the drawbacks of the method. We will extend this discussion in the following section.



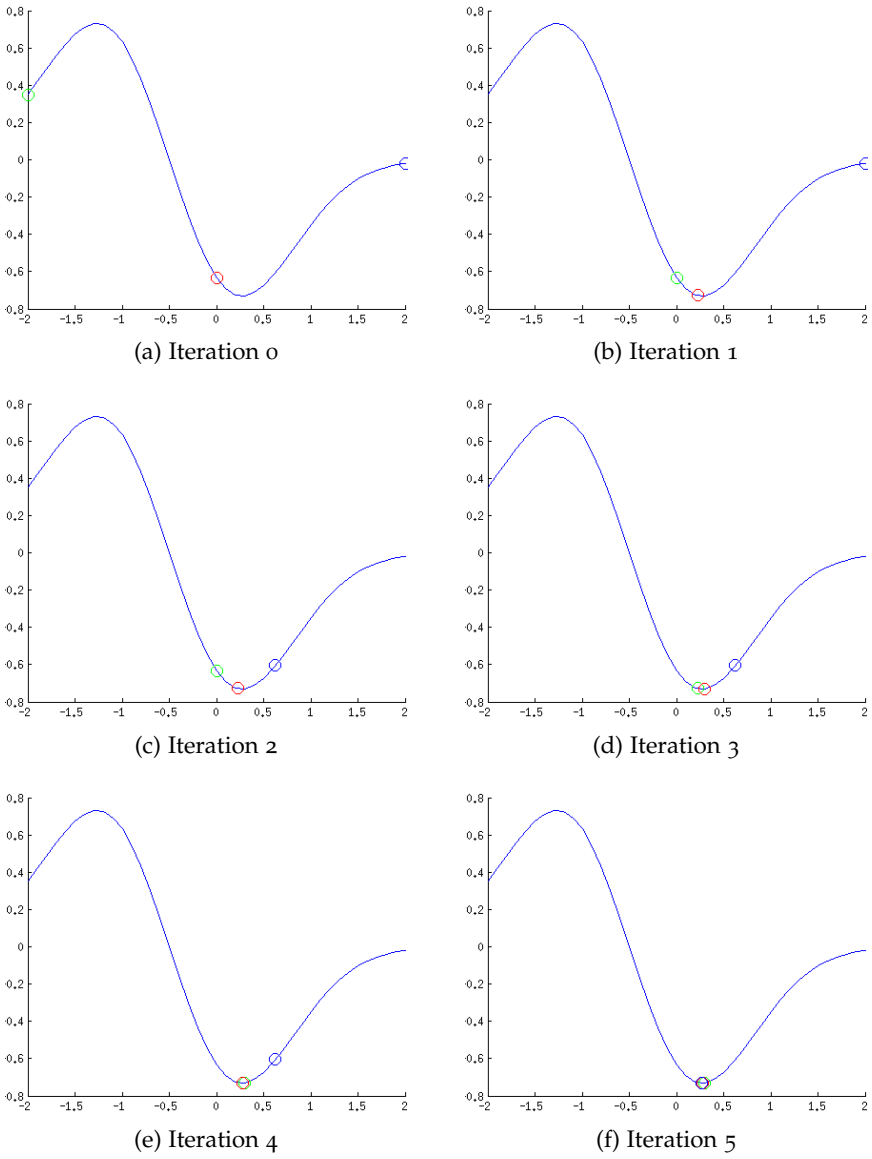
(a) Iteration 0 (b) Iteration 1

(c) Iteration 2 (d) Iteration 3

(e) Iteration 4 (f) Iteration 5

(g) Iteration 6 (h) Iteration 7

Figure 1: Iterations of the Golden Search algorithm for the function $f(x) = -e^{-x^2} + e^{-(x+1)^2}$ given $(-2, 0, 2)$ as initial interval. Green, red and blue are used to show the components of the triplet (a, b, c), respectively.

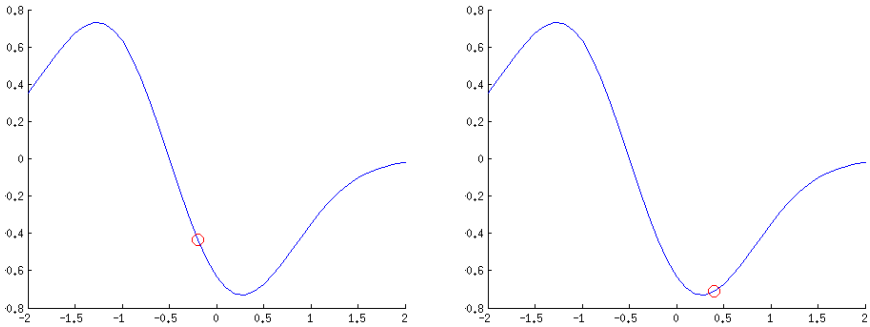


(a) Iteration 0 (b) Iteration 1

(c) Iteration 2 (d) Iteration 3

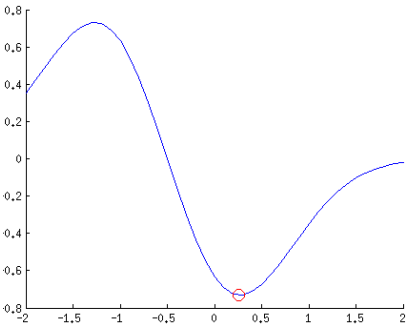
(e) Iteration 4 (f) Iteration 5

Figure 2: Iterations of the Brent's method for the function $f(x) = -e^{-x^2} + e^{-(x+1)^2}$ given $(-2, 0, 2)$ as initial interval. Green, red and blue are used to show the components of the triplet (a, b, c), respectively.



(a) Iteration 0

(b) Iteration 1



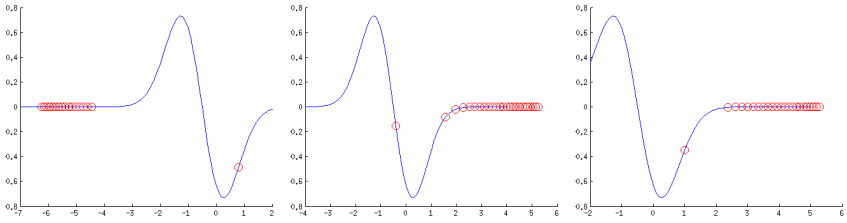
(c) Iteration 2

Figure 3: Iterations of the Gauss-Newton's method for the function $f(x) = -e^{-x^2} + e^{-(x+1)^2}$ with initial value $x_0 = -0.2$.

2.4 COMPARISON OF 1D METHODS

The three methods presented before were evaluated using the function $f(x) = -e^{-x^2} + e^{-(x+1)^2}$ whose minimum is located at $x \approx 0.271702$. In Figures 1, 2 and 3, the iterations performed by the algorithms are presented. In this particular case, the number of iterations performed by Gauss-Newton's method is less than the two other algorithms. However, this value may change according to:

- Initial guess: this case applies to Gauss-Newton's method. The selection of the initial guess is essential because (1) if the it is close to the minimum, the algorithm may converge fast; and (2) if it is close to an inflection point (points in which the concavity of the function changes), the second derivative is close to zero and, hence, the overshooting problem appears. This situation is illustrated in Fig. 4.

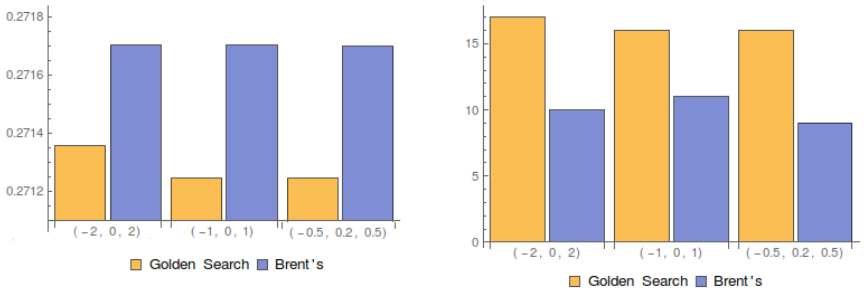


(a) $x_0 = 0.8$ (b) $x_0 = -0.4$ (c) $x_0 = 1$

Figure 4: Overshooting problem of the Gauss-Newton's method for the function $f(x) = -e^{-x^2} + e^{-(x+1)^2}$ different initial guess.

- The size of the initial interval: this case applies to interval-based methods. The larger the interval, the more the algorithm iterates.
- The value of tolerance: the higher the value of tolerance, the more the algorithm iterates. Also, it affects the final result and, hence, it should be chosen properly regarding the type of application in which the algorithm is going to be used.

The last two assertions are exemplified in Fig. 5. Since the region in $[-1, 1]$ is similar to a parabola, Brent's method can approximates easier to the result than Golden Search. Also, the former is more accurate under the same tolerance conditions.



(a) Final result (b) Number of iterations

Figure 5: Final result and number of iterations using the Golden Search and the Brent's method for the function $f(x) = -e^{-x^2} + e^{-(x+1)^2}$. The value of tolerance defined in this case is equal to 1×10^3.

Unlike Gauss-Newton's method, the interval-based algorithms are reliable since they converge after some iterations to the minimum in the given interval. The unique requirement is the a priori knowledge about the function in order to determine the triplet enclosing the minimum.

In Table 5, a summary of important facts discussed so far is presented.

Topic	Golden Search	Brent's Method	Newton's Method
Previous knowledge	interval	interval	derivatives
Derivatives required?	no	no	yes (1st and 2nd order)
Interval required?	yes	yes	no
Convergence	yes	yes	maybe
Number of input	3 (initial triplet)	3 (initial triplet)	1 (initial guess)
Uni-modality?	yes	no	no
Over-shooting	no	no	yes
Reliable?	yes	yes	no

Table 1: Comparison of 1D methods

3

OPTIMISATION OF FUNCTIONS IN TWO DIMENSIONS

In the previous chapter, we discussed about minimisation in one dimension. The main idea of all the algorithms was to find which point to evaluate next by exploring only one direction. Now, in the case of a two dimensional function, the same approach cannot be applied since we are dealing now with a surface and, the directions that can be explored are more. However, we can reduce the problem to a one dimension by considering line search, i.e. given a function f, a current position P and a direction to follow D, the goal is to find the parameter t minimising $f(P + t \cdot D)$.

Summarising, the key questions for n-dimensional approaches is:

- Which direction to follow?
- How far should the algorithm move from the current position?

Both questions are tackled using different strategies. In the following sections, we describe algorithms for n-dimensional optimisation.

3.1 ARBITRARY LINE SEARCH

A naive approach that can be considered is to select arbitrarily two different directions. To illustrate this method we consider the function $f(x, y) = x^2 + y^2$ which minimum is located at $(0, 0)$ (See Fig. 6).

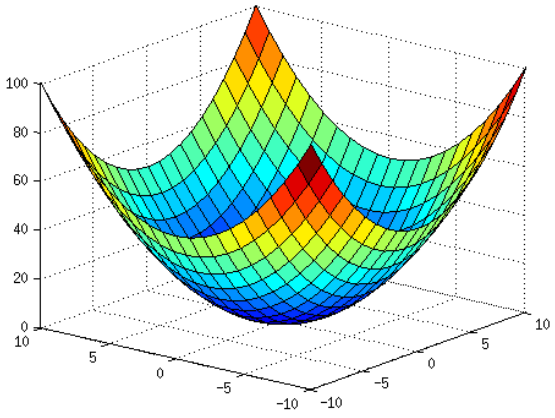
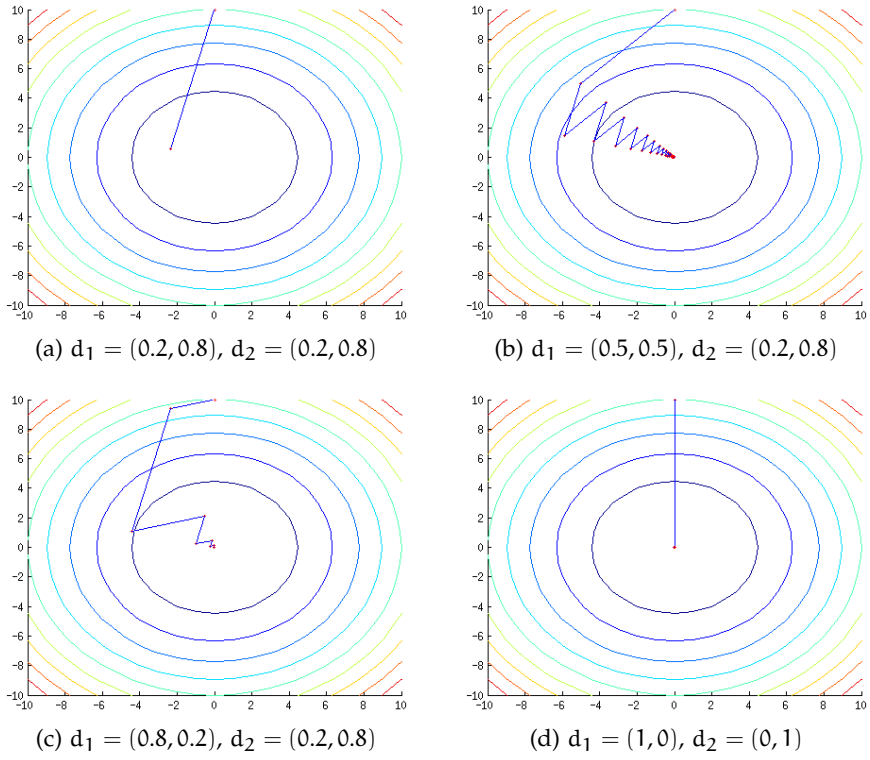


Figure 6: $f(x, y) = x^2 + y^2$

In Fig. 7, we use different directions to show that the selection of the them is essential in order to converge and also to reduce the number



(a) $d_1 = (0.2, 0.8)$, $d_2 = (0.2, 0.8)$

(b) $d_1 = (0.5, 0.5)$, $d_2 = (0.2, 0.8)$

(c) $d_1 = (0.8, 0.2)$, $d_2 = (0.2, 0.8)$

(d) $d_1 = (1, 0)$, $d_2 = (0, 1)$

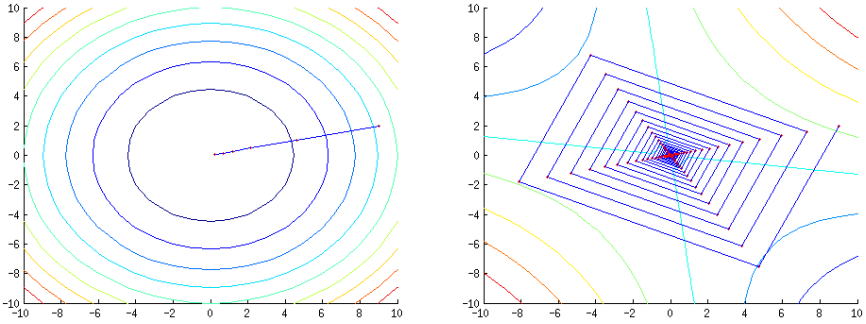
Figure 7: Quadratic function minimization using arbitrary line search. The algorithm converges in (b), (c) and (d) while in (a) it does not.

of iterations. For instace, the algorithm does not converge in Fig 7(a) since the directions are parallel and, thus, the line search is minimising in one way. In general, unless they are pointing to the minimum, parallel directions is not a good option. Furthermore, the number of steps to get to the solution can differ dramatically such as between Fig. 7(b) and Fig. 7(d).

In the following sections, we present methods that cleverly decide the direction for minimising in each step.

3.2 STEEPEST DESCENT METHOD

The Steepest descent method considers the opposite direction of the gradient of the function for determining the next move, i.e to take the direction in which the change is minimum. The approach sounds reasonable since the gradient at the minimum is close or equal to 0. In Fig. 8(a), the algorithm is able to find the minimum of a paraboloid in 1 step (depending on the size of the step). However, the algorithm has a drawback. It is presented in Fig. 8(b). The algorithm start "bouncing" from one level set to another following the direction of the gradient. The new direction after each iteration is orthogonal to the previous one which could lead to follow a wrong line of search.



(a) $f(x,y) = x^2 + y^2$ (b) $f(x,y) = x^2 + 8xy + y^2$

Figure 8: Quadratic function minimization using steepest descent. In (a) the algorithm converges after some iterations depending on the size of the step. In (b) the main drawback of the steepest descent algorithm is presented.

3.3 POWELL'S METHOD

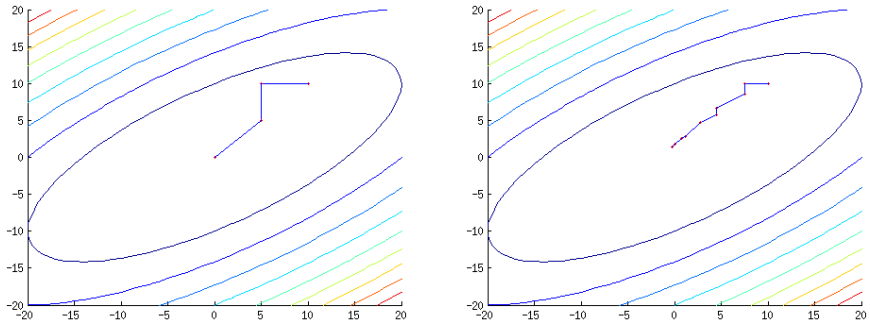
Unlike Steepest descent method, Powell's does not consider the gradient of the function. The method consists basically in selecting two initial vectors, follow their directions and update according to the displacement from the initial point. The Powell's algorithm is presented as follows:

Algorithmus 2 : Powell's method for two dimensions

Data : Initial point P_0)
Result : The minimum of the function P_N

```
n = 0;
u1 = [1 0]ᵀ;
u2 = [0 1]ᵀ;
while is not small enough do
    calculate the point P1 on the direction of u1 from P0;
    calculate the point P2 on the direction of u2 from P1;
    set u1 equal to u2;
    set u2 equal to P2 − P0;
    calculate the point P0 on the direction of u2 from P2;
    set n equal to n + 1;
```

In Fig. 9 the powell's method is used to find the minimum of the function $f(x,y) = 2x^2 - 2xy + y^2$ which is located at $(0,0)$. In this case, we consider different size of the step to illustrate the main disadvantage of the method. The algorithm is able to locate the real minimum in Fig. 9(a), while in Fig. 9(b) does not since, eventually, u_1 and u_2 will tend to be dependant, i.e. $u_1 = \lambda * u_2$ and, hence, the algorithm gives an incorrect answer.



(a) Large step (b) Short step

Figure 9: Quadratic function minimization using powell's method for the function $f(x,y) = 2x^2 - 2xy + y^2$ with different steps. The algorithm is able both cases but the minimum found in (a) is $(0,0)$ while in (b) is approximately equal to $(-0.23, 1.3)$.

3.4 CONJUGATE GRADIENT METHOD

In order to understand Conjugate Gradient method, two features about the previous algorithms should be taken into account:

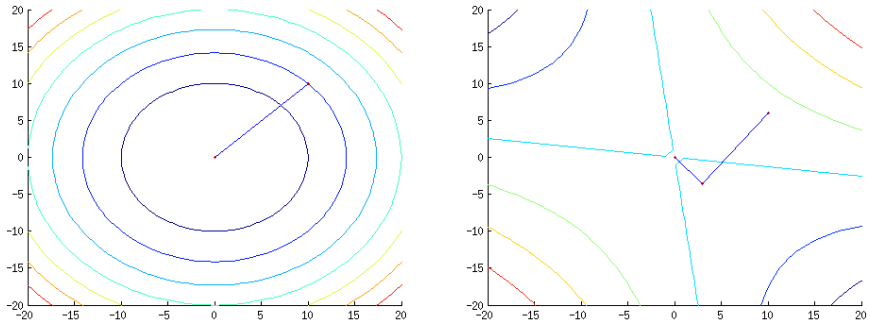
- The Steepest descent method was selecting an orthogonal direction to the previous one in each iteration. However, following the direction of the gradient may cause a "bouncing" effect on the algorithm.
- The main issue of Powell's method, as explained before, is that the directions to follow will tend to be dependent after some iterations.

In order to overcome the previous situations, the Conjugate Gradient method chooses directions which are linearly independent. In fact, a good way to choose the direction is to consider conjugate vectors, i.e. vectors satisfying the following condition:

$$0 = u^T \cdot A \cdot v, \tag{11}$$

where u and v are called conjugate vectors with respect to A. In this way, in each iteration a vector orthogonal to the previous ones is considered.

The Conjugate Gradient method is well-known since it can find the minimum of a quadratic function in less than 3 iterations, i.e. considering only two conjugate gradient vectors. This fact is exemplified in Fig. 10.



(a) $f(x, y) = x^2 + y^2$ (b) $f(x, y) = x^2 + 8xy + y^2$

Figure 10: Quadratic function minimization using Conjugate Gradient method. The algorithm is able to reach the minimum in less than 3 iterations.

3.5 MINIMISATION OF QUADRATIC FORM

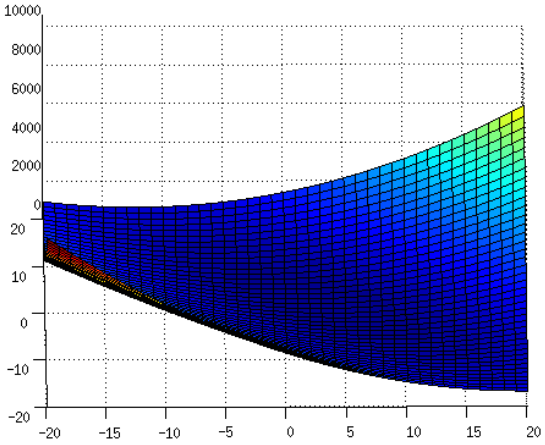
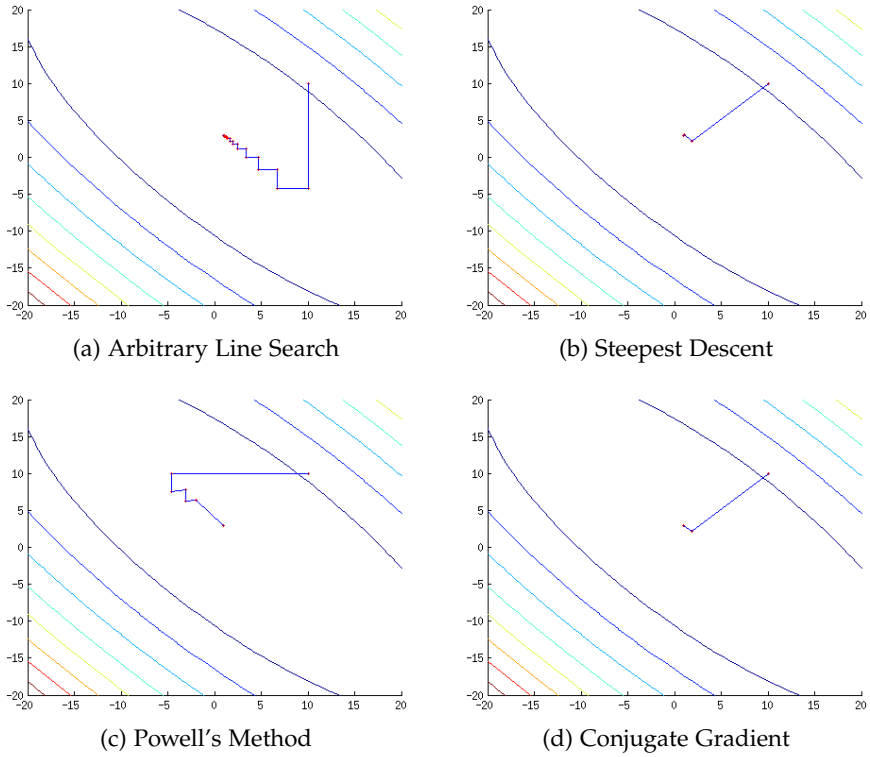


Figure 11: $f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$

The Arbitrary Search using the fundamental basis of $\mathbb{R}^2$, Steepest Descent, Powell's method and Conjugate Gradient were evaluated using a quadratic function called the Booth's function. This function, shown in Fig. 11. has its minimum at $(1, 3)$. The results of the evaluation are graphically presented in Fig. 12 and summarised in Table 2.



(a) Arbitrary Line Search

(b) Steepest Descent

(c) Powell's Method

(d) Conjugate Gradient

Figure 12: Iterations of different algorithms for the Booth's function

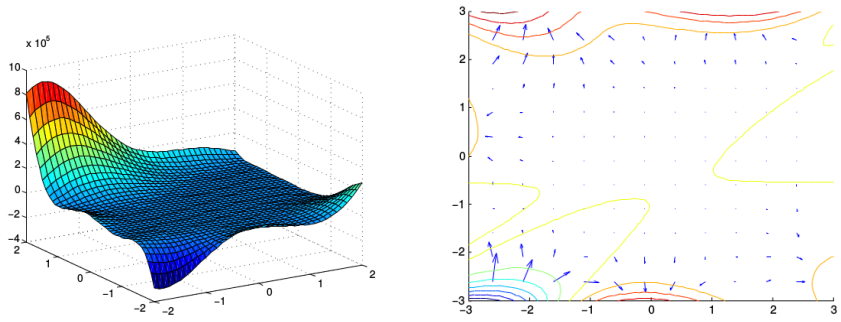
Except for the Arbitrary Search, the methods converge to the expected value. Also, the number of iterations performed by the Arbitrary Search is more than for the other methods. Although, in this particular case, the Steepest Descent approaches the minimum faster, it takes the same number of steps than the Powell's method. The algorithm performing the best in this case is the Conjugate Gradient since, as expected, iterates less than three times.

Method	Iterations	Result	Relative Error
Arbitrary	39	(1.0002, 2.9975)	0.0793%
Steepest Descent	7	(1.0000, 3.0000)	0.0000%
Powell's	7	(1.0000, 3.0000)	0.0000%
Conjugate Gradient	2	(1.0000, 3.0000)	0.0000%

Table 2: Comparison of 2D methods using the Booth's function

3.6 MINIMISATION OF NON-QUADRATIC FORM

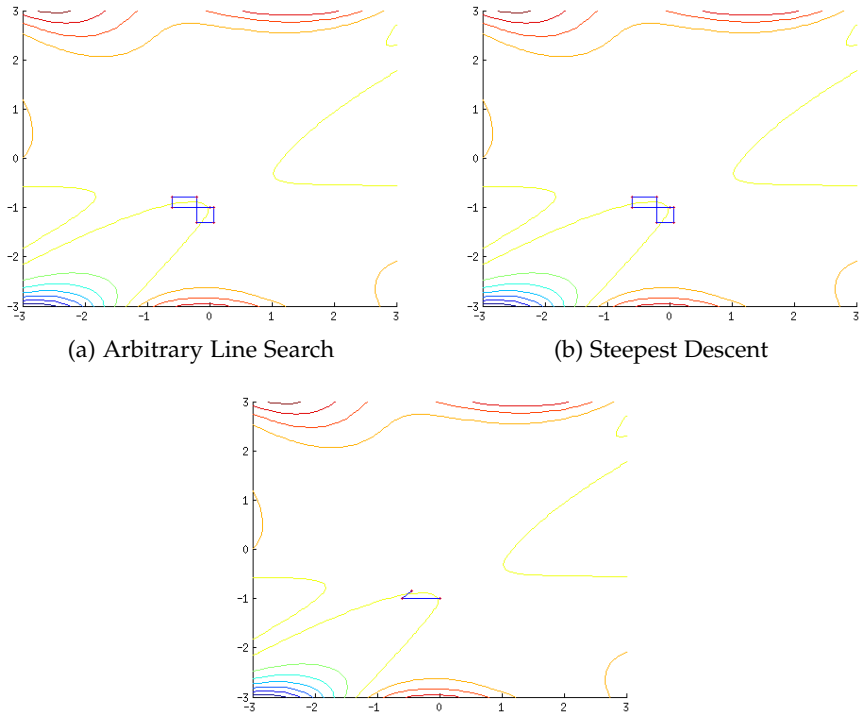
The Goldstein function, presented in Fig. 13, is well-known because it has several local minima and a global minimum located at $(0, -1)$. Furthermore, the gradient of points next to the center is close to 0 while in the borders the magnitude of the it is higher.



(a) Surface (b) Contour and gradient direction

Figure 13: Goldstein function

The Arbitrary Search using the fundamental basis of $\mathbb{R}^2$, Steepest Descent and Broyden's method were evaluated using the Goldstein function. The results of the evaluation are graphically presented in Fig. 14 and summarised in Table 3.



(a) Arbitrary Line Search (b) Steepest Descent

(c) Broyden's method

Figure 14: Iterations of different algorithms for the Goldstein function. The initial point was $(0, -1.0001)$.

In this case, two out of three algorithms are not able to converge to the expected value. Since the two methods are considering orthogonal directions in each iteration the algorithm "bounce" between level sets as seen in Fig. 14(a-b). However, it can be seen that they get close to

the global minimum. On the other hand, Broyden's method converges but the value does not correspond to the real point.

Except for the Arbitrary Search, the methods converge to the expected value. Also, the number of iterations performed by the Arbitrary Search is more than for the other methods. Although, in this particular case, the Steepest Descent approaches the minimum faster, it takes the same number of steps than the Powell's method. The algorithm performing the best in this case is the Conjugate Gradient since, as expected, iterates less than three times.

Method	**Iterations**	**Result**	**Relative Error**
Arbitrary	∞	(0.0625, -0.9965)	6.2598%
Steepest Descent	∞	(0.0629, -0.9967)	6.2987%
Broyden's	6	(-0.4437, -0.8431)	47.0586%

Table 3: Comparison of 2D methods using the Goldstein function

4

OPTIMISATION OF FUNCTIONS IN MULTIPLE DIMENSIONS

4.1 GROUND-TRUTH DATA CREATION

4.2 DATA FITTING USING LEVENBERG-MARQUART

Part III

APPENDIX

A

APPENDIX TEST

Aliquam lectus. Vivamus leo. Quisque ornare tellus ullamcorper nulla. Mauris porttitor pharetra tortor. Sed fringilla justo sed mauris. Mauris tellus. Sed non leo. Nullam elementum, magna in cursus sodales, augue est scelerisque sapien, venenatis congue nulla arcu et pede. Ut suscipit enim vel sapien. Donec congue. Maecenas urna mi, suscipit in, placerat ut, vestibulum ut, massa. Fusce ultrices nulla et nisl.

Etiam ac leo a risus tristique nonummy. Donec dignissim tincidunt nulla. Vestibulum rhoncus molestie odio. Sed lobortis, justo et pretium lobortis, mauris turpis condimentum augue, nec ultricies nibh arcu pretium enim. Nunc purus neque, placerat id, imperdiet sed, pellentesque nec, nisl. Vestibulum imperdiet neque non sem accumsan laoreet. In hac habitasse platea dictumst. Etiam condimentum facilisis libero. Suspendisse in elit quis nisl aliquam dapibus. Pellentesque auctor sapien. Sed egestas sapien nec lectus. Pellentesque vel dui vel neque bibendum viverra. Aliquam porttitor nisl nec pede. Proin mattis libero vel turpis. Donec rutrum mauris et libero. Proin euismod porta felis. Nam lobortis, metus quis elementum commodo, nunc lectus elementum mauris, eget vulputate ligula tellus eu neque. Vivamus eu dolor.

A.1 APPENDIX SECTION TEST

Nulla in ipsum. Praesent eros nulla, congue vitae, euismod ut, commodo a, wisi. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Aenean nonummy magna non leo. Sed felis erat, ullamcorper in, dictum non, ultricies ut, lectus. Proin vel arcu a odio lobortis euismod. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Proin ut est. Aliquam odio. Pellentesque massa turpis, cursus eu, euismod nec, tempor congue, nulla. Duis viverra gravida mauris. Cras tincidunt. Curabitur eros ligula, varius ut, pulvinar in, cursus faucibus, augue.

More dummy text

Nulla mattis luctus nulla. Duis commodo velit at leo. Aliquam vulputate magna et leo. Nam vestibulum ullamcorper leo. Vestibulum condimentum rutrum mauris. Donec id mauris. Morbi molestie justo et pede. Vivamus eget turpis sed nisl cursus tempor. Curabitur mollis sapien condimentum nunc. In wisi nisl, malesuada at, dignissim sit amet, lobortis in, odio. Aenean consequat arcu a ante. Pellentesque porta elit sit amet orci. Etiam at turpis nec elit ultricies imperdiet. Nulla facilisi. In hac habitasse platea dictumst. Suspendisse

LABITUR BONORUM PRI NO	QUE VISTA	HUMAN
fastidii ea ius	germano	demonstratea
suscipit instructior	titulo	personas
quaestio philosophia	facto	demonstrated

Table 4: Autem usu id.

Listing 1: A floating example

```
for i:=maxint to 0 do
begin
{ do nothing }
end;
```

viverra aliquam risus. Nullam pede justo, molestie nonummy, scelerisque eu, facilisis vel, arcu.

A.2 ANOTHER APPENDIX SECTION TEST

Curabitur tellus magna, porttitor a, commodo a, commodo in, tortor. Donec interdum. Praesent scelerisque. Maecenas posuere sodales odio. Vivamus metus lacus, varius quis, imperdiet quis, rhoncus a, turpis. Etiam ligula arcu, elementum a, venenatis quis, sollicitudin sed, metus. Donec nunc pede, tincidunt in, venenatis vitae, faucibus vel, nibh. Pellentesque wisi. Nullam malesuada. Morbi ut tellus ut pede tincidunt porta. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam congue neque id dolor.

Donec et nisl at wisi luctus bibendum. Nam interdum tellus ac libero. Sed sem justo, laoreet vitae, fringilla at, adipiscing ut, nibh. Maecenas non sem quis tortor eleifend fermentum. Etiam id tortor ac mauris porta vulputate. Integer porta neque vitae massa. Maecenas tempus libero a libero posuere dictum. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aenean quis mauris sed elit commodo placerat. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Vivamus rhoncus tincidunt libero. Etiam elementum pretium justo. Vivamus est. Morbi a tellus eget pede tristique commodo. Nulla nisl. Vestibulum sed nisl eu sapien cursus rutrum.

BIBLIOGRAPHY

[1] Greg Fasshauer. Introduction to computational mathematics. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.296.7190&rep=rep1&type=pdf, 2011. [Online; accessed 2015-21-12].

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

http://code.google.com/p/classicthesis/

Happy users of `classicthesis` usually send a real postcard to the author, a collection of postcards received so far is featured here:

http://postcards.miede.de/

DECLARATION

Put your declaration here.

Le Creusot, France, December 2015

Jose Bernal, December 24, 2015