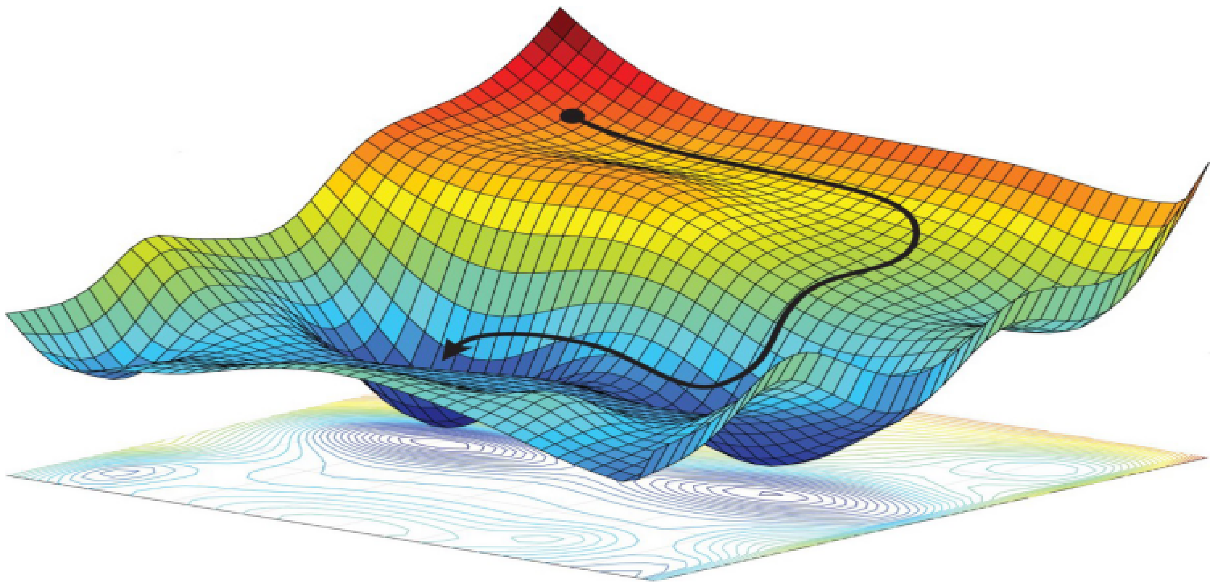
École d'ingénieurs Télécom Physique
Université de Strasbourg

Practical Session Report
Optimization

Thomas DUMOND
Quentin FRERE



École d'ingénieurs
Télécom Physique
Université de Strasbourg

DUMOND Thomas
FRERE Quentin

Summary

DUMOND Thomas
FRERE Quentin

Part 1 : Finding BFS

1. Bourrin Method

For the first part, what was the most complicated was to test the different feasible solutions. We had to understand the purpose of using the function nchoosek. Then we decided to create a list with the feasible combinations. The LP_Bourrin function tests all feasible solutions with the function nchoosek.
We call the function with this line during the test.

```
[f, x, B] = LP_bourrin(A, b, c)
```

2. Simplex Method

In the LP_Simplex function we use the base that is given by the user to test the feasible solutions.
We call the function with this line during the test.

```
[f, x, B] = LP_simplex(A, b, c, v)
```

3. Simplex Two Phases Method

In the LP_Two_Phase_Simplex we complete the original matrix with another base, adding the identity matrix on the right for the first phase. For the second phase, we reuse the simplex method by calling LP_simplex.
We call the function with this line during the test.

```
[f, x, B] = LP_Two_Phase_Simplex(A, b, c)
```

4. Tests

For the tests we used the results of the Q4 from the TD. We implemented the tests of the different functions in the part1_tests.m file. For the tests we used :

```
A = [10 30 30 1 0 0; 1 1 1 0 1 0; 0 1 0 0 0 1];
b = [2500 120 30]';
c = [-474 -774 -645 0 0 0]';
```

And for all these functions we find the same results that are the correct results from the TD solution. So we can expect that the functions are well implemented but we only test it on one example. In addition, we have implemented in all the different functions a test for the arguments to test if they correspond to what it's expected :

DUMOND Thomas
FRERE Quentin

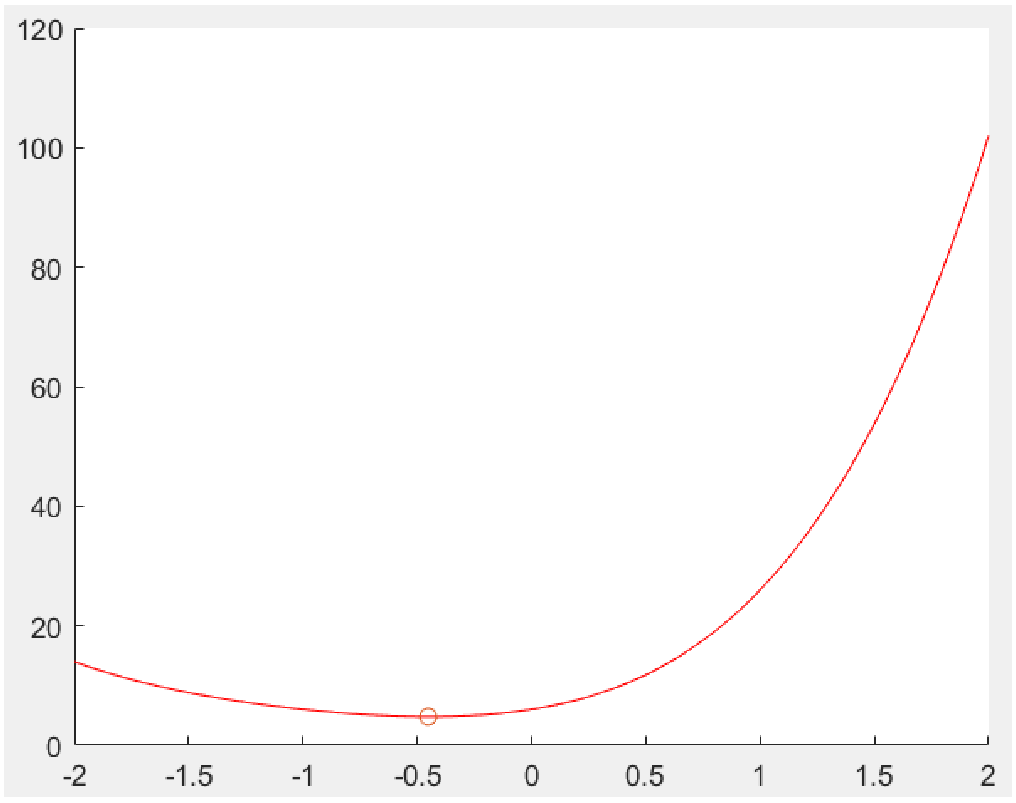
```
f =

  -7.1865e+04


x =

   55.0000
   30.0000
   35.0000
         0
         0
         0


B =

     1     2     3
```

Part 2 : Search of a minimum of functions

1. Golden Section

Then, we implemented the routine using the golden section. We used the indications from what we saw in the magistral course. The idea is to find two points that are distant from $\epsilon = 10^{-}2$. The idea is to compare the function on two points and to reduce the distance between the two points according to the results of the comparison and we subdivide the interval in two by using the golden ratio : $\varphi = \frac{3-\sqrt{5}}{2}$.
We found x in $[-\ 0.4520; \quad -\ 0,4444]$ and for the minimum of $f = \ 4,7991$.

```
while abs(b0 - a0) > eps
    if f(a1) <= f(b1)
        b0 = b1;
    end

    if f(a1) > f(b1)
        a0 = a1;
    end

    a1 = ro * (b0 - a0) + a0;
    b1 = b0 - ro * (b0 - a0);
end
```



2. Newton's Method

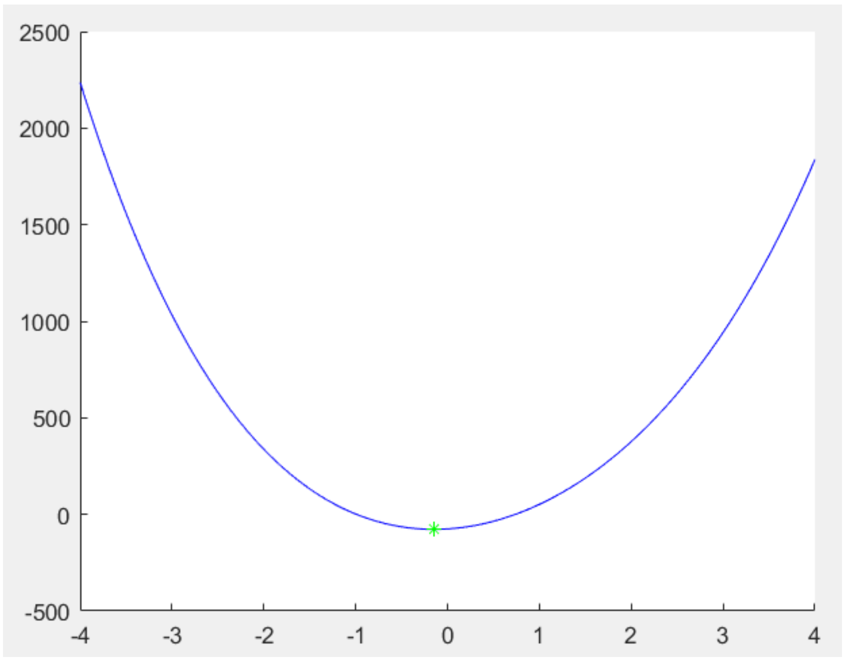
We implemented Newton's method. We didn't use the derivator tool from matlab, we derived the function ourselves, twice. The idea is to use the properties of the derived function to find the appropriate point. This method tries to find the point where the derived function is the most near to zero keeping the derived twice function positive.

```
function xk = Newton(x0, eps, Nmax)
    xk = x0;
    gk = 8 * x0^3 - 15 * x0^2 + 200 * x0 + 30;
    hk = 24 * x0^2 - 30 * x0 + 200;
    n = 0;
    while abs(gk) >= eps && n <= Nmax && hk > 0
        xk = xk - (gk / hk);
        gk = 8 * xk^3 - 15 * xk^2 + 200 * xk + 30;
        hk = 24 * xk^2 - 30 * xk + 200;
        n = n + 1;
    end

end
```

3. Secant Method

About the secant method, we had more problems implementing the routine. We had some troubles using the different variables. The idea is quite the same as the previous method but instead of using the derived twice function, we use the rate that tends to this function.

```
function xk = Secant(x0, x1, eps, Nmax)
    xl = x0;
    xk = x1;
    gl = 8 * x0^3 - 15 * x0^2 + 200 * x0 + 30;
    gk = 8 * x1^3 - 15 * x1^2 + 200 * x1 + 30;
    n = 0;
    while abs(gk) >= eps && n <= Nmax && (gk - gl) / (xk - xl) > 0
        xr = xk - ((xk - xl) / (gk - gl)) * gk;
        xl = xk;
        xk = xr;
        gl = gk;
        gk = 8 * xk^3 - 15 * xk^2 + 200 * xk + 30;
        n = n + 1;
    end
end
```



Both with the Newton method and the secant method we found $x = -0,1482$.

DUMOND Thomas
FRERE Quentin

Part 3 : Gradient method for minimum

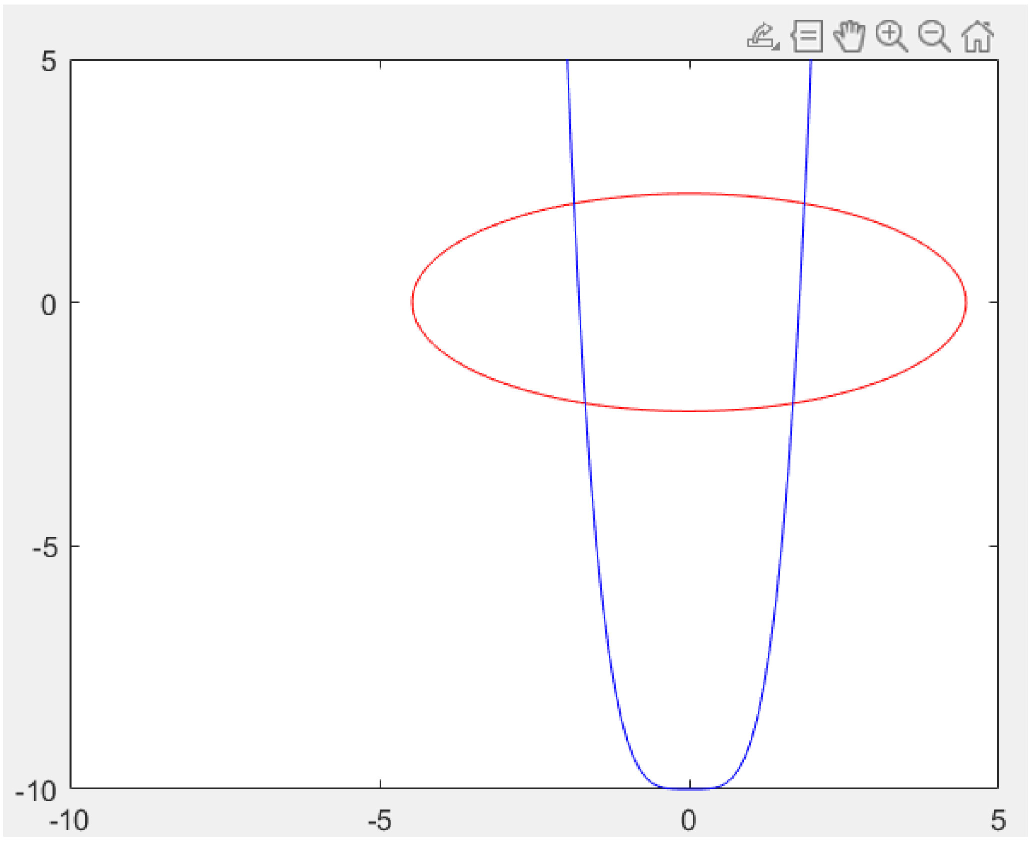
For the third part, we applied the gradient ourselves. The idea of the steepest descent is to minimize the gradient using an iterative method to approximate an integral to find finally an approximation of a stationary point. The results are :

```
with x0=[-0.5 0.5]
x=
   -0.7071    0.0006

with x0=[0.5 -0.5]
x=
   -0.7071   -0.0006

with x0=[1 1]
x=
    1.9160    2.5255
```

Part 4 : Plotting Curves

We plot the curve for the question 4.2 :

DUMOND Thomas
FRERE Quentin


5
0
-5
-10
-10
-5
0
5