Tehnici de Optimizare: Lab 4

Universitatea Aurel Vlaicu Arad

Instructions: The codes should be written in Python. The use of notebooks is encouraged and you may start from the examples given on Moodle.

Preliminary questions. Before going into the exercises make sure you master the following aspects (see the notebook on "Higher dimensional aspects" on Moodle)

- Define and plot a 2D function and its gradient. Contour plot is useful for visualization.
- Compute the eigenvalues and eigenvectors of a square matrix. Test if a matrix is positive definite and compute its condition number.
- Compute various decompositions of a matrix using routines from `numpy.linalg`: Choleski, LU, etc.
- Check if the formula for the gradient of a function is correct by using finite differences

Exercise 1. Gradient descent - Quadratic case

1. A general quadratic function is defined by

$$f(x) = \frac{1}{2}x^T Ax - b \cdot x.$$

The gradient of this function is classical, but you should make sure you understand its computation. Compute the Frechet derivative and compare it to the gradient vector. The matrix form is preferred since it is shorter. Recall however what is the explicit formula for $f(x)$ in terms of (x_i) and the elements of A and b.

Prove that if A is symmetric, positive-definite then the problem

$$\min_{x \in \mathbb{R}^n} f(x)$$

admits a unique solution. Write the associated optimality condition.

2. Study the example given on Moodle, where the gradient descent algorithm with fixed step and some other variants are implemented. Understand how you should modify the two functions, the one which computes the value of the objective function and another one which computes the gradient. In the quadratic case the solution should solve the system $Ax = b$, which you can compute using `numpy.linalg.solve`, in order to test the efficiency of your algorithm. Test your algorithm in the following cases:

(a) $A = \begin{pmatrix} 1 & 0.4 \\ 0.4 & 2 \end{pmatrix}$, $b = [1, 1]$

(b) $A = \begin{pmatrix} 0.78 & -0.02 & -0.12 & -0.14 \\ -0.02 & 0.86 & -0.04 & 0.06 \\ -0.12 & -0.04 & 0.72 & -0.08 \\ -0.14 & 0.06 & -0.08 & 0.74 \end{pmatrix}$, $b = [0.76, 0.08, 1.12, 0.68]$

(c) **(Challenge)** The Hilbert matrix defined by $H_{i,j} = 1/(i + j - 1)$, $1 \leq i, j \leq n$ for $n \in \{4, 8, 16\}$. Prove that H is positive definite and compute its condition number in Python. (**Hint:** you can use the explicit formula for $x^T Hx$ and note that it comes from the integral of a square) Then use the numerical algorithm to find the solution and observe the precision of the computations compared to the condition number of the matrix.

(d) Choose a badly conditioned matrix like $A = \begin{pmatrix} 2000 & 0 \\ 0 & 0.1 \end{pmatrix}$, $b = 0$ and a starting point $x_0 = [-0.5, 1.5]$. Comment the behavior of the optimization algorithm in this case.

Advice:

- Establish a maximal number of iterations in order to avoid infinite loops if something goes wrong. Learn how to stop an infinite loop running if you do not know how to do it!
- As an exit criterion you may take the relative decrease in the value of the function, but this will leave you far from the solution if the function varies slowly. A better criterion is to look at the norm of the gradient and stop when it is small enough.

3. Implement the gradient descent with a line-search procedure of your choice (Armijo, Goldstein-Price, Wolfe).

Test the algorithm on the same functions as above and verify its convergence properties. Compare the number of iterations and function evaluations between this algorithm and the one using a fixed step.

4. In the quadratic case compute the condition number of the matrix and deduce, the expected convergence ratio $\left(\frac{Q-1}{Q+1}\right)$, in terms of the norm $\|\cdot\|_A$. Deduce an upper bound on the number of iterations needed to reach a certain precision. Here you may take two points of view (as shown in the Course notes):

- precision with respect to the function value, i.e. in the norm $\|\cdot\|_A$.
- precision with respect to the position of the minimizer

Compare the theoretical predictions with the number of iterations in each case.

Answers: 1. Using standard linear algebra computations you can find that

$$f(x+h) = f(x) + (Ax-b)\cdot h + \frac{1}{2}h^T Ah.$$

This shows that the Frechet derivative is

$$h \mapsto (Ax-b)\cdot h,$$

and the gradient of f is $Ax-b$.

2. Just use the notebooks given. Look at the notebook for higher dimensional aspects to see (if you don't already know) how to define matrices, compute eigenvalues, condition number, etc. The gradient descent notebook contains the GD algorithm with fixed, variable and optimal step.

3. **Hint**: You may use the code that you produced for the previous session in the 1D case. All you need to do is to include the line-search procedure of your choice in the GD code.

4. Compute Q using the tools given in the introductory notebook regarding the eigenvalues. Alternatively, you may also use the command `numpy.linalg.cond`.

Exercise 2. The Rosenbrock problem

In this exercise we consider the function

$$f(x, y) = 100(y - x^2)^2 + (1 - x)^2.$$

This is a classical example, used for bench-marking optimization software. The function has a unique critical point x^*, the function is non-convex and rather badly conditioned.

1. Compute the gradient ∇f and the Hessian $D^2 f$ for the function defined above. Prove that the function has a unique minimizer and find the minimizer x^*.

2. Run the algorithms developed in the previous exercise in order to approximate numerically the minimizer of f. Choose as starting point $x_0 = (-1.2, 1.2)$. You may vary the position of x_0 if you want.

3. Observe the behavior of the trajectory (x_n) given by the gradient descent when you vary the line-search parameters m_1, m_2. Interpret the observed behavior and decide some "heuristics" on what is the better choice by observing the behavior of the algorithm, the number of iterations, of function evaluations, etc.

4.Denote by H the matrix $\begin{pmatrix} 802 & -400 \\ -400 & 200 \end{pmatrix}$. Compute its eigenvalues and decide if it is positive-definite or not. Towards the end of the algorithm, when you are close to the minimum, replace the descent direction $d = -\nabla f(x)$ by $d = -H^{-1}\nabla f(x)$. Is this still a descent direction?

Hopefully, you should observe that the algorithm converges straight away. Can you explain why?

Answers:

1. Just note that f is a sum of squares, therefore $f \geq 0$. Moreover, if $f = 0$ then the two squares should be equal to 0, giving $x = 1, y = 1$. This shows that there is a unique minimizer.

For the gradient vector just compute the partial derivatives of f with respect to x and y:

$$\frac{\partial f}{\partial x} = 200(y - x^2)(-2x) - 2(1 - x),$$

$$\frac{\partial f}{\partial y} = 200(y - x^2).$$

Please check yourself and simplify. For the Hessian matrix, differentiate the partial derivatives w.r.t. x and y. You may check that the Hessian matrix is symmetric and that $D^2 f(1, 1)$ is positive definite.

2. Normally you should only need to add the Rosenbrock function to your existing algorithm.

3. The matrix H is positive definite: you should check this! A direction d is a descent direction at x if $d \cdot \nabla f(x) < 0$. In our case

$$d \cdot \nabla f(x) = -H^{-1}\nabla f(x) \cdot \nabla f(x) = -(\nabla f(x))^T H^{-1}\nabla f(x).$$

Since H^{-1} is also positive definite (why?) the above computation shows that d is still a descent direction.

In order to observe the connection between H and the Rosenbrock function you may evaluate the Hessian matrix at $(1, 1)$. The algorithm obtained is similar to the Newton algorithm that will be seen in the next session, with the particularity that only the Hessian at the optimum is used.

Exercise 3. Minimal paths again (Bonus)

The objective in this exercise is multiple:

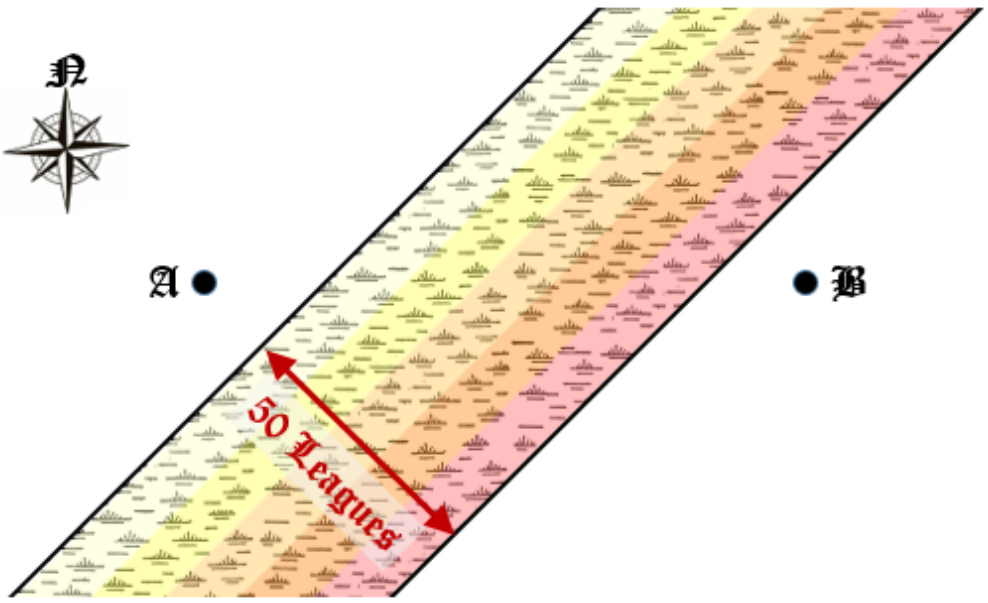
- read the problem and understand what it asks
- translate the ideas into mathematical context
- choose the right mathematical tools to deal with the problem
- arrive at an explicit formula for the function you want to minimize and compute its gradient
- use an appropriate numerical algorithm to find the solution

1. (Path of a light beam) It is known that the light always follows the fastest path when traveling between two points. However, its speed is influenced by the medium in which it travels. Suppose that the speed of light in is $v_a = 299700000$ meters per second. It is known that light travels slower in water and the speed of light in water is about 1.3 smaller than v_a.

Suppose you have a light bulb at the point $(0, 1)$ and the water level is the line $y = 0$. Draw the path of the light from $(0, 1)$ to various points on the line $y = -1$.

A person looks to the water from the point $(0, 1)$ and looks at an object modeled by the segment $[1, 2] \times \{-1\}$. What is the apparent position of the object at the surface of the water?

2. ("Marsh Crossing") This problem is taken from the Project Euler website `https://projecteuler.net/problem=607`. Frodo and Sam need to travel 100 leagues due East from point A to point B. On normal terrain, they can cover 10 leagues per day, and so the journey would take 10 days. However, their path is crossed by a long marsh which runs exactly South-West to North-East, and walking through the marsh will slow them down. The marsh is 50 leagues wide at all points, and the mid-point of AB is located in the middle of the marsh. A map of the region is shown in the diagram below:



The marsh consists of 5 distinct regions, each 10 leagues across, as shown by the shading in the map. The strip closest to point A is relatively light marsh, and can be crossed at a speed of 9 leagues per day. However, each strip becomes progressively harder to navigate, the speeds going down to 8, 7, 6 and finally 5 leagues per day for the final region of marsh, before it ends and the terrain becomes easier again, with the speed going back to 10 leagues per day.

If Frodo and Sam were to head directly East for point B, they would travel exactly 100 leagues, and the journey would take approximately 13.4738 days. However, this time can be shortened if they deviate from the direct path.

Find the shortest possible time required to travel from point A to B, and give your answer in days, rounded to 10 decimal places.

Remark: If you find the problem hard to understand, consider a simplified case where the speed outside the marsh is 10 leagues per day and the speed in the marsh is 5 leagues per day.

Hints:

a) What is the shortest path when traveling between two points with constant speed?

b) How many variables do you need in your problem? (Think of how many points in the plane you need and then decide.)

c) In the picture the marsh is tilted 45° with respect to the horizontal. How can you choose the coordinates system so that your work becomes easier?

d) **(Optional)** Prove that the solution exists and is unique by showing that the function to be minimized is strictly convex.

e) Deduce a formula for the time of travel between A and B in terms of the variables you have chosen. Compute the gradient with respect to all of the variables in your problem.

f) Plug the function and the gradient you obtained into the algorithm developed previously and find the solution. You may also plot the optimal path.

g) **(Optional)** Consider the case where the marsh is divided into $N > 0$ equal strips each of which has equal crossing speed such that the N speeds form an equidistant division of the interval $[5, 10]$. Make a conjecture on what happens when $N \to \infty$.

Answers:

1. Denote by v_a the speed in air and v_w the speed in water. Then if the light bulb is $B(0, 1)$ and the other end point os $C(x_0, -1)$, denoting by $A_x(x, 0)$ a point in the surface of the water, the total time spent by the light on the path BA_xC is

$$\frac{BA_x}{v_a} + \frac{A_xC}{v_w}.$$

It is not difficult to see that the above quantity can be minimized w.r.t. x and the optimal position of A_x depends only on the ratio v_a/v_w.

2. a) Of course if the speed is constant, the straight line gives the shortest time.

b) You need a point for each line where the speed changes: 6 points in total. Each point has two coordinates, but is constrained to remain on a straight line.

c) Instead of working on the given configuartion, rotate everything 45 degrees, so that in the new situation the y coordinates of the variable points are fixed. This reduces the number of variables to 6.

d) The cost function obtained depends on the distance function in the plane which is convex w.r.t. variations of the endpoints of the segments.

e) Simple computation

d) Notebook.

e) This could be handled easily if you automatize the writing of the cost function (use a loop, don't write everything down by hand...).

Exercise 4. Explicit computation of the optimum

This is a supplementary exercise and you should tackle it only if you are interested and only **after doing Exercises 1, 2 and the first part of Exercise 3**. No bonus points will be awarded if you solve this exercise but not the first two and a half!

It was shown briefly in the course that there exist inequalities which allow you to solve explicitly some particular optimization problems even in higher dimensions. Some of these inequalities are recalled below:

- $x^2 \geq 0$: the most basic inequality
- **AM-GM**:
$$x_i \geq 0 \Rightarrow \frac{x_1 + ... + x_n}{n} \geq (x_1 ... x_n)^{1/n}$$
with equality if and only if $x_1 = ... = x_n$.
- **Generalized AM-GM** (or just convexity of the $-\log$ function):
$$x_i > 0, a_i \geq 0, \sum_{i=1}^{n} a_i = 1 \Longrightarrow x_1^{a_1} ... x_n^{a_n} \leq a_1 x_1 + ... + a_n x_n$$
with equality if and only if $x_1 = ... = x_n$.
- **Cauchy-Schwarz**: $a_i, b_i \in \mathbb{R}$
$$\left(\sum_{i=1}^{n} a_i b_i \right)^2 \leq \left(\sum_{i=1}^{n} a_i^2 \right) \left(\sum_{i=1}^{n} b_i^2 \right) \text{ or } |\mathbf{a} \cdot \mathbf{b}| \leq |\mathbf{a}||\mathbf{b}|$$
with equality if and only if $\mathbf{a}$ and $\mathbf{b}$ are colinear.
- **Hölder's inequality**: $(\mathbf{a} \cdot \mathbf{b}) \leq \|\mathbf{a}\|_p \|\mathbf{b}\|_q$ with $p, q > 1$ and $\frac{1}{p} + \frac{1}{q} = 1$. Recall that
$$\|\mathbf{a}\|_p = \left(\sum_{i=1}^{n} a_i^p \right)^{1/p}.$$
The equality takes place if and only if $\mathbf{a}$ and $\mathbf{b}$ are colinear.

In each of the cases below it is possible to find the solution explicitly using one of the above inequalities.

1. minimize $f(x, y) = \dfrac{12}{x} + \dfrac{18}{y} + xy$ on $(0, \infty)^2$
2. maximize $f(x, y) = xy(72 - 3x - 4y)$
3. minimize $f(x, y) = 4x + \dfrac{x}{y^2} + \dfrac{4y}{x}$ on $(0, \infty)^2$
4. maximize $f(x, y, z) = 2x + 3y + 6z$ when $x^2 + y^2 + z^2 = 1$
5. maximize $f(x, y, z) = 2x + 3y + 6z$ when $x^p + y^p + z^p = 1$, $p > 1$.

Hints:

1. AM-GM
2. AM-GM
3. AM-GM
4. Cauchy-Schwarz
5. Hölder