Operations Research III: Theory

Implementation of Gradient Descent

Jenny Tzeng

Department of Information Management
National Taiwan University

Road map

- **Gradient descent**.
- Gradient descent for convex quadratic functions.

Gradient descent

- We introduced the **gradient descent** method to solve nonlinear programs in *Operations Research II: Algorithms*.
- We have discussed about the importance of convex analysis.
- Gradient descent may result in a sub-optimal solution if the objective function is non-convex.
- In general, implementing gradient descent may be hard.
- In this video, we try to implement gradient descent for two convex quadratic functions with Python.
  - The implementation does not work for other types of functions.

Steps to implement gradient descent

- Before we see the demo file, we review the steps to describe gradient descent algorithm which was mentioned in *Operations Research II: Algorithms*.
  - Step 0: Choose an initial point x^0 and a precision parameter $\varepsilon > 0$.
  - Step $k + 1$:
    - Find $\triangledown f(x^k)$.
    - Solve $a_k = \text{argmin}_{a \geq 0} \ f(x^k - a \triangledown f(x^k))$.
    - Update the current solution to $x^{k+1} = x^k - a_k \triangledown f(x^k)$,
    - If $||\triangledown f(x^{k+1})|| < \epsilon$, stop; otherwise let k become $k + 1$ and continue.
  - Because our function f is a convex quadratic function:
    - Its gradient may be obtained analytically.
    - The optimization for the step size is to solve another single-variate convex quadratic function.

Road map

- Gradient descent.
- **Gradient descent for convex quadratic functions**.

Example 1

- Let's solve

$$\min f(x) = x_1^2 + 2x_1 - 2x_1x_2 + 2x_2^2.$$

- We have

$$\nabla f(x) = (2x_1 - 2x_2 + 2, -2x_1 + 4x_2).$$

- The Hessian matrix is

$$\begin{vmatrix} 2 & -2 \\ -2 & 4 \end{vmatrix},$$

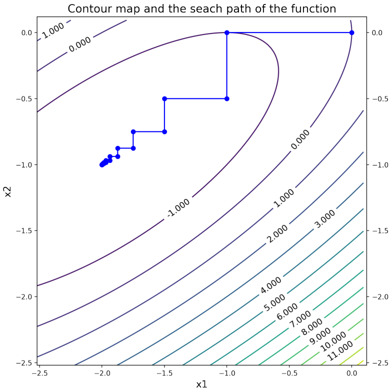
  and the two leading principal minors are 2 and 4. The function is convex.
- According to FOC, we have the optimal solution $x^* = (-2, -1)$, where $f(x^*) = -2$.
- Let's see how to use gradient descent to solve it with Python.

Example 1: solve and interpret

- In this instance, we choose $x^0 = (0, 0)$ and $\varepsilon = 0.00001$.
- The algorithm stops at the 36^{th} iteration, and the solution is exactly the same.

Example 1: solve and interpret

- We illustrate the function and the search path.
  - From the contour map, we figure out the shape of this quadratic function with two variables is an ellipse.
  - We start the algorithm from $x^0 = (0, 0)$; the next solution is $x^1 = (-1, 0)$. It keeps searching until the norm of the gradient is less than the precision parameter.



Example 2

- Let's solve

$$\min f(x) = 4x_1^2 - 4x_1 + 2x_1x_2 - 20x_2 + 5x_2^2 + 2.$$

- We have

$$\nabla f(x) = (8x_1 + 2x_2 - 4, 2x_1 + 10x_2 - 20).$$

- The Hessian matrix is

$$\begin{vmatrix} 8 & 2 \\ 2 & 10 \end{vmatrix},$$

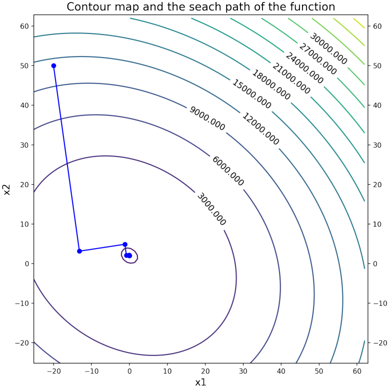
  and the two leading principal minors are 8 and 76. The function is convex.
- According to FOC, we have the optimal solution $x^* = (0, 2)$, where $f(x^*) = -18$.
- Let's see how to use gradient descent to solve it with Python.

Example 2: solve and interpret

- In this instance, we choose $x^0 = (-20, 50)$ and $\varepsilon = 0.00001$.
- The algorithm stop at the 13^{th} iteration, and the solution is exactly the same.

Example 2: solve and interpret

- We illustrate the function and the search path.
  - The shape of this function is also an ellipse because the shape of a convex quadratic function with two variables must be an ellipse.
  - From search paths of these two examples,we may observe that the angles are **right angles**. This is a feature of gradient descent if the step size is chosen to be the most improving one.



Some Remarks

- Gradient descent could be used to solve more than quadratic functions with two variables.
  - However, more complicated implementations are needed.
- **Convex analysis is important**. If a problem is not convex, we may end up with a local minimum instead of a global minimum.