Non-Linear Optimization Methods - Part III

Optimization Methods in Management Science
Master in Management
HEC Lausanne

Dr. Rodrigue Oeuvray

Fall 2019 Semester

Non-Linear Optimization Methods

Unconstrained optimization:

- Newton method with linesearch
- Quasi-Newton methods
- BFGS algorithm

Newton Method With Linesearch

- A globally convergent algorithm is an algorithm that converges to a local optimum from **any** initial point (not to be confused with global optimization !)

- The Newton's method is **not globally convergent** but we can combine a **linesearch with the Newton's direction** to get a **globally convergent** algorithm

- When the hessian is **not positive definite**, the Newton's direction is not a **descent** direction

- If the hessian is not positive definite, we can slightly modified it to get a matrix satisfying this property

- When it is too **time-expensive** to compute the hessian, then we can use a quasi-Newton method to approximate it

LDL^T Decomposition

- **Cholesky** decomposition of a **positive definite** matrix $\boldsymbol{Q}$:

$$\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$$

- If $\boldsymbol{S}$ is the diagonal matrix containing the main diagonal of $\boldsymbol{L}$, then we can write

$$\boldsymbol{Q} = \boldsymbol{L}_u\boldsymbol{D}\boldsymbol{L}_u^T,$$

  where $\boldsymbol{D} = \boldsymbol{S}^2$ and $\boldsymbol{L}_u = \boldsymbol{L}\boldsymbol{S}^{-1}$

- $\boldsymbol{D}$ is a diagonal matrix with strictly positive elements and $\boldsymbol{L}_u$ is a lower unit triangular matrix. A **unit lower triangular matrix** is a lower triangular matrix with ones on the diagonal

- This decomposition is called the LDL^T decomposition

LDL^T Decomposition: Example

Let's try to find the LDL^T decomposition of the matrix $\boldsymbol{Q}$:

$$\boldsymbol{Q} = \begin{pmatrix} 2 & -1 & 1 \\ -1 & 3 & 0 \\ 1 & 0 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ L_{21} & 1 & 0 \\ L_{31} & L_{32} & 1 \end{pmatrix} \begin{pmatrix} D_1 & 0 & 0 \\ 0 & D_2 & 0 \\ 0 & 0 & D_3 \end{pmatrix} \begin{pmatrix} 1 & L_{21} & L_{31} \\ 0 & 1 & L_{32} \\ 0 & 0 & 1 \end{pmatrix}$$

Now we can easily multiply out the $\boldsymbol{D}$ and $\boldsymbol{L}_u^T$ matrices on the right to get

$$\boldsymbol{Q} = \begin{pmatrix} 2 & -1 & 1 \\ -1 & 3 & 0 \\ 1 & 0 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ L_{21} & 1 & 0 \\ L_{31} & L_{32} & 1 \end{pmatrix} \begin{pmatrix} D_1 & D_1 L_{21} & D_1 L_{31} \\ 0 & D_2 & D_2 L_{32} \\ 0 & 0 & D_3 \end{pmatrix}$$

Now multiplying the first row in $\boldsymbol{L}_u$ times the first column in the second matrix shows that $D_1 = 2$

Multiplying the first row in $\boldsymbol{L}_u$ times the second column gives $D_1 L_{21} = -1$, so $L_{21} = -1/2$

The first row in $\boldsymbol{L}_u$ times the third column gives $D_1 L_{31} = 1$, so $L_{31} = 1/2$

LDL^T Decomposition: Example (Cont'd)

If we fill in what's known, here's how things stand at the moment:

$$\boldsymbol{Q} = \begin{pmatrix} 2 & -1 & 1 \\ -1 & 3 & 0 \\ 1 & 0 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -1/2 & 1 & 0 \\ 1/2 & L_{32} & 1 \end{pmatrix} \begin{pmatrix} 2 & -1 & 1 \\ 0 & D_2 & D_2 L_{32} \\ 0 & 0 & D_3 \end{pmatrix}$$

One can check that the second row in $\boldsymbol{L}_u$ times the first column in the second matrix on the right gives -1

Now multiply second row times second column to get $1/2 + D_2 = 3$, so $D_2 = 5/2$

Second row by third column gives $-1/2 + D_2 L_{32} = 0$, so $L_{32} = (1/2)/D2 = 1/5$

We currently have

$$\boldsymbol{Q} = \begin{pmatrix} 2 & -1 & 1 \\ -1 & 3 & 0 \\ 1 & 0 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -1/2 & 1 & 0 \\ 1/2 & 1/5 & 1 \end{pmatrix} \begin{pmatrix} 2 & -1 & 1 \\ 0 & 5/2 & 1/2 \\ 0 & 0 & D_3 \end{pmatrix}$$

Finally, take the third row times third column to find that $D_3 = 22/5$

LDL^T Decomposition: Example (Cont'd)

The entire decomposition looks like

$$\boldsymbol{Q} = \begin{pmatrix} 2 & -1 & 1 \\ -1 & 3 & 0 \\ 1 & 0 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -1/2 & 1 & 0 \\ 1/2 & 1/5 & 1 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & 5/2 & 0 \\ 0 & 0 & 22/5 \end{pmatrix} \begin{pmatrix} 1 & -1/2 & 1/2 \\ 0 & 1 & 1/5 \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \boldsymbol{L}\boldsymbol{L}^T = \begin{pmatrix} \sqrt{2} & 0 & 0 \\ -1/\sqrt{2} & \sqrt{5}/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{10} & \sqrt{22}/\sqrt{5} \end{pmatrix} \begin{pmatrix} \sqrt{2} & -1/\sqrt{2} & 1/\sqrt{2} \\ 0 & \sqrt{5}/\sqrt{2} & 1/\sqrt{10} \\ 0 & 0 & \sqrt{22}/\sqrt{5} \end{pmatrix}$$

with $\boldsymbol{L} = \boldsymbol{L}_u \boldsymbol{D}^{0.5}$ and

$$\boldsymbol{D}^{0.5} = \begin{pmatrix} \sqrt{2} & 0 & 0 \\ 0 & \sqrt{5}/\sqrt{2} & 0 \\ 0 & 0 & \sqrt{22}/\sqrt{5} \end{pmatrix}$$

Modified Cholesky Factorization

- Suppose that in computing some D_k we find that $D_k \leq 0$
- Then we replace D_k by some fixed number $\delta > 0$ and we continue the factorization as before
- When we're done we have matrices $\boldsymbol{L}_u$ and $\boldsymbol{D}$, and we can form a **positive definite** matrix $\tilde{\boldsymbol{Q}} = \boldsymbol{L}_u \boldsymbol{D} \boldsymbol{L}_u^T$
- $\tilde{\boldsymbol{Q}}$ is different from the original matrix $\boldsymbol{Q}$
- This decomposition is a modified Cholesky decomposition of $\boldsymbol{Q}$

Modified Cholesky Factorization: Remarks

- We can perform this modified Cholesky decomposition to $\nabla^2 f(\boldsymbol{x}_k)$ to get a **positive definite** matrix approximating the hessian

$$\nabla^2 f(\boldsymbol{x}_k) \approx \boldsymbol{L}_u \boldsymbol{D} \boldsymbol{L}_u^T = \boldsymbol{L}\boldsymbol{L}^T,$$

  with $\boldsymbol{L} = \boldsymbol{L}_u \boldsymbol{D}^{0.5}$

- If the hessian is **positive definite** then the modified Cholesky decomposition provides the same decomposition as the Cholesky decomposition

- With this decomposition of the hessian, then we have the guarantee that the direction $\boldsymbol{d}_k$ computed as

$$\boldsymbol{L}\boldsymbol{L}^T \boldsymbol{d}_k = -\nabla f(\boldsymbol{x}_k)$$

  is a **descent** direction

The Newton's Method with Linesearch

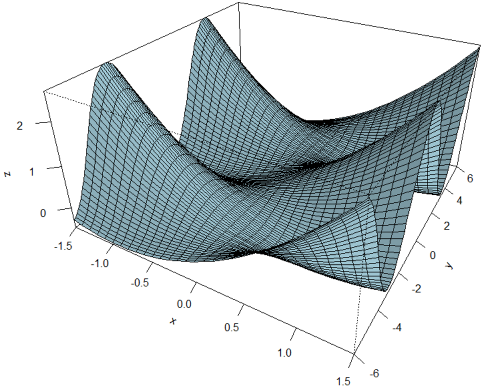
- **Input**: a first approximation $\boldsymbol{x}_0$ of the solution and a convergence parameter ϵ
- **Output**: an approximate solution $\boldsymbol{x}^*$
- **Initialization**: $k = 0$
- **Iterations**:
  - Compute a modified Cholesky factorization $\boldsymbol{L}\boldsymbol{L}^T$ of $\nabla^2 f(\boldsymbol{x}_k)$
  - Solve $\boldsymbol{L}\boldsymbol{L}^T \boldsymbol{d}_k = -\nabla f(\boldsymbol{x}_k)$
    - Determine $\boldsymbol{z}_k$ by solving $\boldsymbol{L}\boldsymbol{z}_k = -\nabla f(\boldsymbol{x}_k)$
    - Determine $\boldsymbol{d}_k$ by solving $\boldsymbol{L}^T \boldsymbol{d}_k = \boldsymbol{z}_k$
  - Determine α_k with a linesearch
  - $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k$
  - $k = k + 1$
- **Stopping criterion**: if $\|\nabla f(\boldsymbol{x}_k)\| \leq \epsilon$, then $\boldsymbol{x}^* = \boldsymbol{x}_k$

The Newton's Method with Linesearch: An Example

We would like to minimize the following function: $\mathbb{R}^2 \rightarrow \mathbb{R}$ defined by

$$f(x_1, x_2) = \frac{1}{2}x_1^2 + x_1 cos x_2$$

The graph of this function is given below:



Example (Cont'd)

- Reminders about $cos(x)$ and $sin(x)$:
  - $cos(0) = 1$, $cos(\pi/2) = 0$, $cos(\pi) =$ -1, $cos(3\pi/2) = 0$
  - $sin(0) = 0$, $sin(\pi/2) = 1$, $sin(\pi) = 0$, $sin(3\pi/2) =$ -1
  - the derivative of $sin(x)$ is $cos(x)$ and of $cos(x)$ is -$sin(x)$

- The gradient of f is given by:

$$\nabla f(x_1, x_2) = \left(\begin{array}{c} x_1 + cos\ x_2 \\ \text{-}x_1 sin\ x_2 \end{array} \right)$$

- On can check that it is null at $\boldsymbol{x}_k^* = ((\text{-}1)^{k+1} \;\; k\pi)^T$ for $k \in \mathbb{Z}$ and at $\bar{\boldsymbol{x}}_k = (0 \;\; \frac{\pi}{2} + k\pi)^T$, $k \in \mathbb{Z}$

Example (Cont'd)

- The hessian of f is given by

$$\nabla^2 f(x_1, x_2) = \begin{pmatrix} 1 & \text{-}sin\ x_2 \\ \text{-}sin\ x_2 & \text{-}x_1 cos\ x_2 \end{pmatrix}$$

- By evaluating the hessian at $\boldsymbol{x}_k^*$, we get:

$$\nabla^2 f(\boldsymbol{x}_k^*) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

- This matrix is postive definite. We conclude that $\boldsymbol{x}_k^* = ((\text{-}1)^{k+1} \quad k\pi)^T$ are **local minima**

Example (Cont'd)

- By evaluating the hessian at $\bar{\boldsymbol{x}}_k$, we get:

$$\nabla^2 f(\bar{\boldsymbol{x}}_k) = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \quad \text{if k is odd}$$

  and

$$\nabla^2 f(\bar{\boldsymbol{x}}_k) = \begin{pmatrix} 1 & -1 \\ -1 & 0 \end{pmatrix} \quad \text{if k is even}$$

- A **necessary** (but not sufficient !) condition to have a local minimum is that the matrix must be **positive semi-definite**. If it is not the case, then it is **not possible** to have a minimum

- It is easy to show that none of these matrices are positive semi-definite. It means that the points $\bar{\boldsymbol{x}}_k$ cannot be local minima

- To summarize, the local minima are given by $\boldsymbol{x}_k^* = ((-1)^{k+1} \quad k\pi)^T$ for $k \in \mathbb{Z}$

Example (Cont'd)

How can we show that the following matrix is not positive semi-definite ?

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

We just need to compute

$$(x \ y) \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = x^2 + 2xy$$

and to note that this expression is strictly negative when $y = \text{-}x \neq 0$

Example (Cont'd)

- We use the algorithm described before with a starting point at $(1\ 1)^T$

k	$f(\boldsymbol{x}_k)$	$\|\nabla f(\boldsymbol{x}_k)\|$
0	$1.04030231e+00$	$1.75516512e+00$
1	$2.34942031e-01$	$8.88574897e-01$
2	$4.21849003e-02$	$4.80063696e-01$
3	$-4.52738278e-01$	$2.67168927e-01$
4	$-4.93913638e-01$	$1.14762780e-01$
5	$-4.99982955e-01$	$5.85174623e-03$
6	$-5.00000000e-01$	$1.94633135e-05$
7	$-5.00000000e-01$	$2.18521663e-10$
8	$-5.00000000e-01$	$1.22460635e-16$

- The solution given by the algorithm is $\boldsymbol{x}^* = (1\ \pi)^T$ for which $f(\boldsymbol{x}^*) =$ -0.5, $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$, and $\nabla^2 f(\boldsymbol{x}^*) = \boldsymbol{I}$. This is a **local minimum** of f
- Note that this algorithm only provides one local optimum while there are many of them

Quasi-Newton Methods versus Newton's Method - 1

Important Remark

Newton's method

- advantage: **fast convergence**
- disadvantages:
  - requires second derivatives which can be too expensive to compute for large scale applications
  - the hessian may be singular
  - the hessian is not necessary a positive definite matrix

Quasi-Newton Methods versus Newton's Method - 2

Important Remark

Quasi-Newton methods:

- the hessian of the objective function is **approximated** using updates based on **gradient** evaluations
- the most common quasi-Newton algorithm is the Broyden-Fletcher- Goldfarb-Shannon (BFGS) algorithm
- the approximation of the hessian is always **positive definite** with the **BFGS** algorithm

BFGS Update - 1

- The **BFGS update** provides an approximation of the **hessian** (not of its inverse !) of the objective function
- It is given by

$$\boldsymbol{H}_k = \boldsymbol{H}_{k-1} + \frac{\boldsymbol{y}_{k-1}\boldsymbol{y}_{k-1}^T}{\boldsymbol{y}_{k-1}^T\bar{\boldsymbol{d}}_{k-1}} - \frac{\boldsymbol{H}_{k-1}\bar{\boldsymbol{d}}_{k-1}\bar{\boldsymbol{d}}_{k-1}^T\boldsymbol{H}_{k-1}}{\bar{\boldsymbol{d}}_{k-1}^T\boldsymbol{H}\bar{\boldsymbol{d}}_{k-1}}$$

with $\bar{\boldsymbol{d}}_{k-1} = \alpha_{k-1}\boldsymbol{d}_{k-1} = \boldsymbol{x}_k - \boldsymbol{x}_{k-1}$ and $\boldsymbol{y}_{k-1} = \nabla f(\boldsymbol{x}_k) - \nabla f(\boldsymbol{x}_{k-1})$

- Note that this approximation only uses information about the **first order** derivates of the objective function

BFGS Update - 2

- This approximation is always **positive definite**
- At each iteration, we need an approximation $\boldsymbol{H}_k^{-1}$ of the **inverse** of the hessian of the objective function in order to be able to compute the following descent direction

$$\boldsymbol{d}_k = -\boldsymbol{H}_k^{-1} \nabla f(\boldsymbol{x}_k)$$

- We need to **inverse** the approximation of the hessian given by the **BFGS update**
- The inverse of a symmetric definite positive matrix is also a symmetric positive definite matrix

BFGS Update - 3

- By applying the **Sherman-Morrison** formula to the BFGS update, then we can derive an approximation of the **inverse** of the hessian of the objective function
- This update is given by the following formula:

$$\boldsymbol{H}_k^{-1} = \left(\boldsymbol{I} - \frac{\bar{\boldsymbol{d}}_{k-1}\boldsymbol{y}_{k-1}^T}{\bar{\boldsymbol{d}}_{k-1}^T\boldsymbol{y}_{k-1}}\right)\boldsymbol{H}_{k-1}^{-1}\left(\boldsymbol{I} - \frac{\bar{\boldsymbol{d}}_{k-1}\boldsymbol{y}_{k-1}^T}{\bar{\boldsymbol{d}}_{k-1}^T\boldsymbol{y}_{k-1}}\right) + \frac{\bar{\boldsymbol{d}}_{k-1}\bar{\boldsymbol{d}}_{k-1}^T}{\bar{\boldsymbol{d}}_{k-1}^T\boldsymbol{y}_{k-1}},$$

with $\bar{\boldsymbol{d}}_{k-1} = \alpha_{k-1}\boldsymbol{d}_{k-1} = \boldsymbol{x}_k - \boldsymbol{x}_{k-1}$ and $\boldsymbol{y}_{k-1} = \nabla f(\boldsymbol{x}_k) - \nabla f(\boldsymbol{x}_{k-1})$

Quasi-Newton's BFGS Method

- **Input**: a convergence parameter $\epsilon > 0$, an initial point $\boldsymbol{x}_0$, and a first approximation of the inverse of the hessian $\boldsymbol{H}_0^{-1}$. It should be a symmetric positive definite matrix. By default, $\boldsymbol{H}_0^{-1} = \boldsymbol{I}$
- **Output**: an approximate solution $\boldsymbol{x}^*$
- **Initialization**: $k = 0$
- **Iterations**:
  - Compute $\boldsymbol{d}_k = -\boldsymbol{H}_k^{-1} \nabla f(\boldsymbol{x}_k)$
  - Determine α_k with a linesearch
  - $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k$
  - $k = k + 1$
  - $\boldsymbol{H}_k^{-1}$ update:

$$\boldsymbol{H}_k^{-1} = \left(\boldsymbol{I} - \frac{\bar{\boldsymbol{d}}_{k-1} \boldsymbol{y}_{k-1}^T}{\bar{\boldsymbol{d}}_{k-1}^T \boldsymbol{y}_{k-1}} \right) \boldsymbol{H}_{k-1}^{-1} \left(\boldsymbol{I} - \frac{\bar{\boldsymbol{d}}_{k-1} \boldsymbol{y}_{k-1}^T}{\bar{\boldsymbol{d}}_{k-1}^T \boldsymbol{y}_{k-1}} \right) + \frac{\bar{\boldsymbol{d}}_{k-1} \bar{\boldsymbol{d}}_{k-1}^T}{\bar{\boldsymbol{d}}_{k-1}^T \boldsymbol{y}_{k-1}},$$

  with $\bar{\boldsymbol{d}}_{k-1} = \alpha_{k-1} \boldsymbol{d}_{k-1} = \boldsymbol{x}_k - \boldsymbol{x}_{k-1}$ and $\boldsymbol{y}_{k-1} = \nabla f(\boldsymbol{x}_k) - \nabla f(\boldsymbol{x}_{k-1})$
- **Stopping criterion**: if $\|\nabla f(\boldsymbol{x}_k)\| \leq \epsilon$, then $\boldsymbol{x}^* = \boldsymbol{x}_k$