Optimization Theory and Applications

Kun Zhu (zhukun@nuaa.edu.cn)

November 21, 2018

Introduction

- Gradient methods use only gradient information (first derivative)
- If higher derivatives are used, the resulting algorithm may perform better (but it may be more computationally demanding)
- Newton's method uses the gradient and the Hessian to determine the search direction
- Newton's method performs better than the steepest descent method if the initial point is close to the minimizer

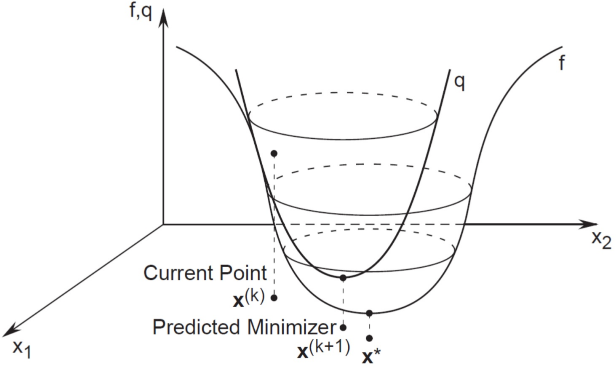
Underlying Idea of Newton's Method

- Given a start point
- Construct a quadratic approximation to the objective function that matches the first and second derivative values at that point
- Minimize the approximate quadratic function instead of the original objective function
- Use the minimizer of the approximate function as the starting point and repeat the procedure iteratively

Newton's Method

- Given: $f : \mathbb{R}^n \to \mathbb{R}$, and current iterate $\mathbf{x}^{(k)}$
- To compute $\mathbf{x}^{(k+1)}$, approximate f by a quadratic

$$q(\mathbf{x}) = f(\mathbf{x}^{(k)}) + (\mathbf{x} - \mathbf{x}^{(k)})^T \mathbf{g}^{(k)} + \frac{1}{2}(\mathbf{x} - \mathbf{x}^{(k)})^T F(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)})$$



Newton's Method

- Use minimizer of q as next iterate $\mathbf{x}^{(k+1)}$
- Write $\mathbf{g}^{(k)} = \nabla f(\mathbf{x}^{(k)})$. By FONC, we have $\nabla q(\mathbf{x}^{(k)}) = 0$

$$\nabla q(\mathbf{x}^{(k)}) = \mathbf{g}^{(k)} + F(\mathbf{x}^{(k)})(\mathbf{x} - \mathbf{x}^{(k)}) = 0$$

- Newton's algorithm

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - F(\mathbf{x}^{(k)})^{-1}\mathbf{g}^{(k)}$$

- Note: no step size (or step size = 1)

Newton's Method

- Example: Use Newton's method to minimize
  $f(x_1,x_2,x_3,x_4) = (x_1+10x_2)^2+5(x_3-x_4)^2+(x_2-2x_3)^4+10(x_1-x_4)^4$
  Starting point $\mathbf{x}^{(0)} = [3,-1,0,1]^T$. Perform three iterations
- For $\mathbf{x}^{(0)}$, $f(\mathbf{x}^{(0)}) = 215$ and

$$\nabla f(\mathbf{x}) = \begin{bmatrix} 2(x_1+10x_2)+40(x_1-x_4)^3 \\ 20(x_1+10x_2)+4(x_2-2x_3)^3 \\ 10(x_3-x_4)-8(x_2-2x_3)^3 \\ -10(x_3-x_4)-40(x_1-x_4)^3 \end{bmatrix}$$

$$F(\mathbf{x}) = \begin{pmatrix} 2+120(x_1-x_4)^2 & 20 & 0 & -120(x_1-x_4)^2 \\ 20 & 200+12(x_2-2x_3)^2 & -24(x_2-2x_3)^2 & 0 \\ 0 & -24(x_2-2x_3)^2 & 10+48(x_2-2x_3)^2 & -10 \\ -120(x_1-x_4)^2 & 0 & -10 & 10+120(x_1-x_4)^2 \end{pmatrix}$$

Newton's Method

- Iteration 1

$$\boldsymbol{g}^{(0)} = [306, -144, -2, -310]^\top,$$

$$\boldsymbol{F}(\boldsymbol{x}^{(0)}) = \begin{bmatrix} 482 & 20 & 0 & -480 \\ 20 & 212 & -24 & 0 \\ 0 & -24 & 58 & -10 \\ -480 & 0 & -10 & 490 \end{bmatrix},$$

$$\boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1} = \begin{bmatrix} 0.1126 & -0.0089 & 0.0154 & 0.1106 \\ -0.0089 & 0.0057 & 0.0008 & -0.0087 \\ 0.0154 & 0.0008 & 0.0203 & 0.0155 \\ 0.1106 & -0.0087 & 0.0155 & 0.1107 \end{bmatrix}$$

$$\boldsymbol{F}(\boldsymbol{x}^{(0)})^{-1}\boldsymbol{g}^{(0)} = [1.4127, -0.8413, -0.2540, 0.7460]^\top.$$

- Hence
  $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} - F(\mathbf{x}^{(0)})^{-1}\mathbf{g}^{(0)} = [1.5873, -0.1587, 0.2540, 0.2540]^T$
  $f(\mathbf{x}^{(1)}) = 31.8$

Newton's Method

- Iteration 2

$$\mathbf{g}^{(1)} = [94.81, -1.179, 2.371, -94.81]^T$$

$$F(\mathbf{x}^{(1)}) = \begin{bmatrix} 215.3 & 20 & 0 & -213.3 \\ 20 & 205.3 & -10.67 & 0 \\ 0 & -10.67 & 31.34 & -10 \\ -213.3 & 0 & -10 & 223.3 \end{bmatrix}$$

$$F(\mathbf{x}^{(1)})^{-1}\mathbf{g}^{(1)} = [0.5291, -0.0529, 0.0846, 0.0846]^T$$

- Hence

$$\mathbf{x}^{(2)} = \mathbf{x}^{(1)} - F(\mathbf{x}^{(1)})^{-1}\mathbf{g}^{(1)} = [1.0582, -0.1058, 0.1694, 0.1694]^T$$

$$f(\mathbf{x}^{(2)}) = 6.28$$

Newton’s Method

- Iteration 3

$$\mathbf{g}^{(2)} = [28.09 - 0.34750.7031 - 28.08]^T$$

$$F(\mathbf{x}^{(2)}) = \begin{bmatrix} 96.8 & 20 & 0 & -94.8 \\ 20 & 202.4 & -4.744 & 0 \\ 0 & -4.744 & 19.49 & -10 \\ -94.80 & 0 & -10 & 104.8 \end{bmatrix}$$

- Hence

$$\mathbf{x}^{(3)} = \mathbf{x}^{(2)} - F(\mathbf{x}^{(2)})^{-1}\mathbf{g}^{(2)} = [0.7037, -0.0704, 0.1121, 0.1111]^T$$

$$f(\mathbf{x}^{(2)}) = 1.24$$

Newton's Method

- The kth iteration of Newton's method can be break down into two steps:
  - Solve $F(\mathbf{x}^{(k)})\mathbf{d}^{(k)} = -\mathbf{g}^{(k)}$ for $\mathbf{d}^{(k)}$
  - Set $\mathbf{x}^{(k+1)} = \mathbf{x}^{k} + \mathbf{d}^{(k)}$
- Step 1 requires the solution of an $n \times n$ system of linear equations
- An efficient method for solving systems of linear equations is essential when using Newton's method

Analysis of Newton's Method

- Does the method work? When does it work? How well does it work?
- For general f
  - Hessian may not be invertible
  - Algorithm may not converge if we do not start close enough to $\mathbf{x}^*$
  - It may not have descent property
  - If it works, it is fast

Analysis of Newton's Method

- If f is a quadratic (with invertible Hessian Q), then Newton's method always converges to $\mathbf{x}^*$ in 1 step. The order of convergence is ∞
- For $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x} - \mathbf{x}^T\mathbf{b}$, the gradient and the Hessian are

$$\mathbf{g}(\mathbf{x}) = \nabla f(\mathbf{x}) = Q\mathbf{x} - \mathbf{b}$$

$$F(\mathbf{x}) = Q$$

Hence, given any initial point $\mathbf{x}^{(0)}$, by Newton's method

$$\begin{aligned}
\mathbf{x}^{(1)} &= \mathbf{x}^{(0)} - F(\mathbf{x}^{(0)})^{(-1)}\mathbf{g}^{(0)} \\
&= \mathbf{x}^{(0)} - Q^{-1}[Q\mathbf{x}^{(0)} - \mathbf{b}] \\
&= Q^{-1}\mathbf{b} \\
&= \mathbf{x}^*
\end{aligned}$$

Convergence of Newton's Method

- What is the order of convergence of Newton's method for general f
- ***Theorem***: Suppose that $f \in \mathcal{C}^3$ and $\mathbf{x}^* \in \mathbb{R}^n$ is a point such that $\nabla f(\mathbf{x}^*) = 0$ and $F(\mathbf{x}^*)$ is invertible. Then for all $\mathbf{x}^{(0)}$ sufficiently close to $\mathbf{x}^*$, Newton's method is well-defined for all k and converges to $\mathbf{x}^*$ with an order of convergence at least 2
- Idea of proof: show $\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\| = O(\|\mathbf{x}^{(k)} - \mathbf{x}^*\|^2)$

  Thus

$$\lim_{k\to\infty} \frac{\|\mathbf{x}^{(k+1)}-\mathbf{x}^*\|}{\|\mathbf{x}^{(k)}-\mathbf{x}^*\|^2} = \lim_{k\to\infty} \frac{O(\|\mathbf{x}^{(k)}-\mathbf{x}^*\|)}{\|\mathbf{x}^{(k)}-\mathbf{x}^*\|^2} = 0$$

Newton's Method and Descent Property

- Note that in the above Theorem, we did not state that $\mathbf{x}^*$ is a local minimizer
- If $\mathbf{x}^*$ is a local maximizer, and if $f \in \mathcal{C}^3$ and $F(\mathbf{x}^*)$ is invertible, Newton's method would converge to $\mathbf{x}^*$ if we start close enough
- Newton's method may not have descent property
  - It is possible that $f(\mathbf{x}^{(k+1)}) > f(\mathbf{x}^{(k)})$
- Fortunately, the vector $\mathbf{d}^{(k)} = -F(\mathbf{x}^{(k)})^{-1}\mathbf{g}^{(k)})$

  points in a direction of decreasing f

Newton's Method and Descent Property

- ***Theorem***: Let $\{\mathbf{x}^{(k)}\}$ be the sequence generated by Newton's method for minimizing a given objective function $f(\mathbf{x})$. If the Hessian $F(\mathbf{x}^{(k)}) > 0$ and $\mathbf{g}^{(k)} = \nabla f(\mathbf{x}^{(k)}) \neq 0$, then the search direction

$$\mathbf{d}^{(k)} = -F(\mathbf{x}^{(k)})^{-1}\mathbf{g}^{(k)} = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}$$

from $\mathbf{x}^{(k)}$ to $\mathbf{x}^{(k+1)}$ is a descent direction for f in the sense that there exists an $\bar{\alpha} > 0$ such that for all $\alpha \in (0, \bar{\alpha})$

$$f(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)}) < f(\mathbf{x}^{(k)})$$

Newton's Method and Descent Property

- Proof:
  - Let $\phi(\alpha) = f(\mathbf{x}^{(k)} + \alpha \mathbf{d}^{(k)})$
  - Then using the chain rule, we obtain

    $\phi'(\alpha) = \nabla f(\mathbf{x}^{(k)} + \alpha \mathbf{d}^{(k)})^T \mathbf{d}^{(k)}$
  - Hence, due to $F(\mathbf{x}^{(k)})^{-1} > 0$ and $\mathbf{g}^{(k)} \neq 0$,

    $\phi'(0) = \nabla f(\mathbf{x}^{(k)})^T \mathbf{d}^{(k)} = -\mathbf{g}^{(k)T} F(\mathbf{x}^{(k)})^{-1} \mathbf{g}^{(k)} < 0$
  - Thus, there exists an $\bar{\alpha} > 0$ so that for all $\alpha \in (0, \bar{\alpha})$, $\phi(\alpha) < \phi(0)$. This implies that for all $\alpha \in (0, \bar{\alpha})$

    $$f(\mathbf{x}^{(k)} + \alpha \mathbf{d}^{(k)}) < f(\mathbf{x}^{(k)})$$

Modification of Newton's Method

- It is possible to modify the algorithm such that the descent property holds
- The above theorem motivates the following modification of Newton's method:

  $$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k F(\mathbf{x}^{(k)})^{-1}\mathbf{g}^{(k)}$$

  where

  $$\alpha_k = \arg\min_{\alpha \geq 0} f(\mathbf{x}^{(k)} - \alpha_k F(\mathbf{x}^{(k)})^{-1}\mathbf{g}^{(k)})$$

- At each iteration, we perform a line search in the direction $-F(\mathbf{x}^{(k)})^{-1}\mathbf{g}^{(k)}$
- Similar to the steepest descent method

Modification of Newton's Method

- Drawbacks of Newton's method
  - Evaluation of $F(\mathbf{x}^{(k)})$ for large n can be computationally expensive
  - Solve the set of n linear equations $F(\mathbf{x}^{(k)})\mathbf{d}^{(k)} = -\mathbf{g}^{(k)}$
  - Which one is more time consuming
- Another potential issue is that the Hessian matrix may not be positive definite
- Why?

Modification of Newton's Method

- If the Hessian $F(\mathbf{x}^{(k)})$ is not positive definite, then the search direction $\mathbf{d}^{(k)} = -F(\mathbf{x}^{(k)})^{-1}\mathbf{g}^{(k)}$ may not point in descent direction
- Is there any way to address this problem
- ***Levenberg-Marquardt modification*** of Newton's method: a simple technique to ensure that the search direction is a descent direction

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (F(\mathbf{x}^{(k)}) + \mu_k \mathbf{I})^{-1}\mathbf{g}^{(k)}$$

  where $\mu_k \geq 0$

Modification of Newton's Method

- The main idea:
  - For a symmetric matrix F which may not be positive definite
  - Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of F with corresponding eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$
  - The eigenvalues are real but may not be positive
  - Next, consider the matrix $G = F + \mu\mathbf{I}$, where $\mu > 0$. Accordingly, the eigenvalues of G is $\lambda_1 + \mu, \ldots, \lambda_n + \mu$

$$\begin{aligned} G\mathbf{v}_i &= (F + \mu\mathbf{I})\mathbf{v}_i \\ &= F\mathbf{v}_i + \mu\mathbf{I}\mathbf{v}_i \\ &= \lambda_i\mathbf{v}_i + \mu\mathbf{v}_i \\ &= (\lambda_i + \mu)\mathbf{v}_i \end{aligned}$$

  - If μ is sufficiently large, all eigenvalues of G are positive and G is positive definite. Accordingly, the search direction $(F(\mathbf{x}^{(k)}) + \mu_k\mathbf{I})^{-1}\mathbf{g}^{(k)}$ always points in a descent direction

Modification of Newton's Method: Levenberg-Marquardt Modification

- Furthermore, we can also introduce a step size α_k as follows:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k (F(\mathbf{x}^{(k)}) + \mu_k \mathbf{I})^{-1} \mathbf{g}^{(k)}$$

- When $\mu_k \to 0$, it approaches the behavior of the pure Newton's method
- When $\mu_k \to \infty$, it approaches a pure gradient method with small step size
- In practice, we can start with a small value of μ and increase it slowly until we find that the iteration is descent

Newton's Method for Nonlinear Least Squares

- We now examine a particular class of optimization problems and the use of Newton's method for solving them. Consider the following problem

  $$\min \ \sum_{i=1}^{m}(r_i(\mathbf{x}))^2$$

  where $r_i(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ are given functions

- This particular problem is called a ***nonlinear least-squares problem***

- A special case, r_i is linear

Newton's Method for Nonlinear Least Squares

- Defining $\mathbf{r} = [r_1, \ldots, r_m]^T$, we write the objective function as
  $$f(\mathbf{x}) = \mathbf{r}(\mathbf{x})^T \mathbf{r}(\mathbf{x})$$
- To apply Newton's method, we first compute the gradient and the Hessian of f. The jth component of $\nabla f(\mathbf{x})$ is
  $$\nabla f(\mathbf{x})_j = \frac{\partial f}{\partial x_j}(\mathbf{x}) = 2\sum_{i=1}^{m} r_i(\mathbf{x}) \frac{\partial r_i}{\partial x_j}(\mathbf{x})$$
- Denote the Jacobian matrix of $\mathbf{r}$ by
  $$\boldsymbol{J}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial r_1}{\partial x_1}(\boldsymbol{x}) & \cdots & \frac{\partial r_1}{\partial x_n}(\boldsymbol{x}) \\ \vdots & & \vdots \\ \frac{\partial r_m}{\partial x_1}(\boldsymbol{x}) & \cdots & \frac{\partial r_m}{\partial x_n}(\boldsymbol{x}) \end{bmatrix}$$
- Then the gradient of f can be represented as
  $$\nabla f(\mathbf{x}) = 2J(\mathbf{x})^T \mathbf{r}(\mathbf{x})$$

Newton's Method for Nonlinear Least Squares

- Next, we compute the Hessian matrix of f, The jth component of the Hessian is given by

$$
\begin{aligned}
\frac{\partial^2 f}{\partial x_k \partial x_j}(\boldsymbol{x}) &= \frac{\partial}{\partial x_k}\left(\frac{\partial f}{\partial x_j}(\boldsymbol{x})\right) \\
&= \frac{\partial}{\partial x_k}\left(2\sum_{i=1}^{m} r_i(\boldsymbol{x})\frac{\partial r_i}{\partial x_j}(\boldsymbol{x})\right) \\
&= 2\sum_{i=1}^{m}\left(\frac{\partial r_i}{\partial x_k}(\boldsymbol{x})\frac{\partial r_i}{\partial x_j}(\boldsymbol{x}) + r_i(\boldsymbol{x})\frac{\partial^2 r_i}{\partial x_k \partial x_j}(\boldsymbol{x})\right)
\end{aligned}
$$

- Let $S(\mathbf{x})$ be the matrix with the (k,j)th component as

$$\sum_{i=1}^{m} r_i(\mathbf{x})\frac{\partial^2 r_i}{\partial x_k \partial x_j}(\mathbf{x})$$

- We write the Hessian matrix as

$$F(\mathbf{x}) = 2(J(\mathbf{x})^T J(\mathbf{x}) + S(\mathbf{x}))$$

Newton's Method for Nonlinear Least Squares

- Therefore, Newton's method applied to the nonlinear least-squares problem is given by

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (J(\mathbf{x})^T J(\mathbf{x}) + S(\mathbf{x}))^{-1} J(\mathbf{x})^T \mathbf{r}(\mathbf{x})$$

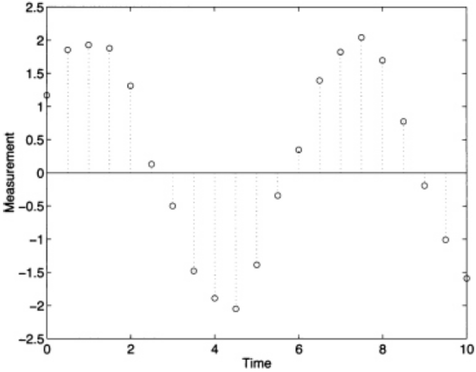
- Note
  - In some applications, the matrix $S(\mathbf{x})$ involving the second derivatives can be ignored because its components are negligibly small. In this case, the Newton's method reduces to what is commonly called Gauss-Newton method

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (J(\mathbf{x})^T J(\mathbf{x}))^{-1} J(\mathbf{x})^T \mathbf{r}(\mathbf{x})$$

  - Note that the Gauss-Newton method does not require calculation of the second derivatives of $\mathbf{r}$

Newton's Method for Nonlinear Least Squares

- ***Example***: Suppose we are given m measurements of a process at m points in time. Let $t_1, \ldots, t_m$ denote the measurement times and $y_1, \ldots, y_m$ the measurement values. We wish to fit a sinusoid to the measurement data



Newton's Method for Nonlinear Least Squares

- The equation of the sinusoid is

  $$y = A \sin(\omega t + \phi)$$

  with appropriate choices of the parameters A, ω, and ϕ

- To formulate the data-fitting problem, we construct the objective function

  $$\sum_{i=1}^{m} (y_i - A \sin(\omega t + \phi))^2$$

  representing the sum of the squared errors between the measurement values and the function values at the corresponding points

Newton's Method for Nonlinear Least Squares

- Let $\mathbf{x} = [A, \omega, \phi]^T$ represent the vector of decision variables. We therefore obtain a nonlinear least-squares problem as

$$r_i(\mathbf{x}) = y_i - A\sin(\omega t + \phi)$$

- The Jacobian matrix $J(\mathbf{x})$ is given by

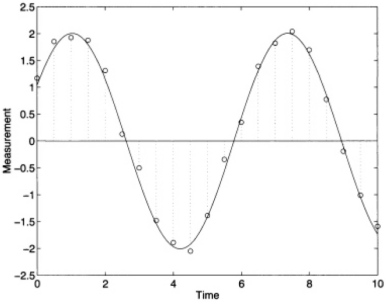
$$(J(\mathbf{x}))_{(i,1)} = -sin(\omega t_i + \phi)$$

$$(J(\mathbf{x}))_{(i,2)} = -t_i Acos(\omega t_i + \phi)$$

$$(J(\mathbf{x}))_{(i,3)} = -Acos(\omega t_i + \phi)$$

Newton's Method for Nonlinear Least Squares

- Using the expressions above, we apply the Gauss-Newton method to find the sinusoid of best fit. The parameters of this sinusoid are $A = 2.01$, $\omega = 0.992$, and $\phi = 0.541$



Summary of Newton's Method

- Newton's method performs well if we start close enough
- We can incorporate a step size to ensure descent
- For a quadratic, converges in one step
- Is there some way of using only gradients, but still only converge in one or a finite number of steps for quadratics?
- Yes. Conjugate direction method