Lecture 23: Newton's Method

Zizhuo Wang

Institute of Data and Decision Analytics (iDDA)
Chinese University of Hong Kong, Shenzhen

Nov 28, 2018

Announcement

- Homework 8 due today (11/28)
- Homework 9 posted, due on the Wednesday after next week (Dec 12)
- No class on Friday (11/30)
- No office hour today (11/28), office hour will be 2pm-4pm tomorrow (11/29)

Recap

We have introduced several methods for unconstrained optimization problems

For one-dimensional problems

- Bisection method and golden section method

For high-dimensional problems, we first studied the gradient descent method

- Choose direction as the negative of the gradient
- Stepsize: Exact line search and backtracking line search (preferred in practice)
- Stopping criterion: Based on the norm of the gradient
- Convergence: Linear convergence
- Other properties: Globally convergence, perpendicular search direction when using exact line search

Pros and Cons of Gradient Descent Method

Pros:

- Easy to understand and implement
- Only need to know the first-order (gradient) information
- Globally convergent, doesn't depend on the initial point

Cons:

- Convergence speed may not be fast enough: Linear convergence

Newton's Method

We introduced Newton's method for solving one dimensional problem:

- Solve $g(x) = f'(x) = 0$.

Newton's method to solve equation:

- Approximate the function by a linear function and then solve the root of the linear function
- That is, use the sequence:

$$x^{k+1} = x^k - \frac{g(x^k)}{g'(x^k)}$$

- In Newton's method, we assume $g'(x) \neq 0$ in the search region.

Convergence of Newton's Method (for 1-Dimension Case)

Theorem

If $g(x)$ is twice continuously differentiable and x^* is a root of $g(x)$ at which $g'(x^*) \neq 0$, then provided that $|x^0 - x^*|$ is sufficiently small, the sequence generated by the Newton iterations:

$$x^{k+1} = x^k - \frac{g(x^k)}{g'(x^k)}$$

will satisfy

$$|x^{k+1} - x^*| \leq C|x^k - x^*|^2$$

with $C = \sup_x \frac{1}{2}|\frac{g''(x)}{g'(x)}|$.

- We call this convergence speed *quadratic convergence*—much faster then linear convergence

Back to the Optimization Problem

Remember $g(x) = f'(x)$ where $f(x)$ is the function we want to minimize.

Therefore, in terms of $f(\cdot)$, the Newton iteration can written as:

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)}$$

With proper conditions, this sequence of $\{x^k\}$ converges to a KKT point

- When $f(\cdot)$ is convex, it converges to the global minimizer (under proper conditions)

Connection to the Gradient Descent Method

In Newton's method:

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)}$$

Remember in the gradient descent method:

$$x^{k+1} = x^k - \alpha f'(x^k)$$

Therefore in the one dimension case, Newton's method simply specifies a unique step size in the gradient method (rather than doing line searches).

- In high-dimensional case, however, Newton's method will also alter the direction.

Another View of Newton's Method

Consider the function $f(x)$ we want to minimize. We first write the second-order Taylor expansion at current step x^k:

$$f(x) \approx f(x^k) + f'(x^k)(x - x^k) + \frac{1}{2} f''(x^k)(x - x^k)^2$$

What is the minimizer of the right hand side term?

- Solve the quadratic function, we get the minimizer is:

$$x^k - \frac{f'(x^k)}{f''(x^k)}$$

  which is exactly the next iterate in Newton's method.

- Therefore one can view Newton's method as first approximating the objective function by a quadratic function locally, then minimize that quadratic function.
- If the original objective function is a quadratic function, then Newton's method converges in 1 step.
- This idea is useful to study high-dimensional case.

Newton's Method in High Dimensional Case

We minimize $f(\mathbf{x})$ where $\mathbf{x}$ is a n-dimensional vector.

At $\mathbf{x}^k$, we approximate the objective function by its second order Taylor expansion:

$$f(\mathbf{x}) \approx f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T(\mathbf{x} - \mathbf{x}^k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^k)^T \nabla^2 f(\mathbf{x}^k)(\mathbf{x} - \mathbf{x}^k)$$

We minimize this approximation function and get

$$\mathbf{x}^* = \mathbf{x}^k - (\nabla^2 f(\mathbf{x}^k))^{-1} \nabla f(\mathbf{x}^k)$$

Therefore, we define the search direction (called Newton's step)

$$\mathbf{d}^k = -(\nabla^2 f(\mathbf{x}^k))^{-1} \nabla f(\mathbf{x}^k).$$

In the gradient descent method, the search direction is $-\nabla f(\mathbf{x}^k)$.

- Therefore, Newton's method refines the search direction by using the second-order information $\nabla^2 f(\mathbf{x}^k)$

Newton's Method in High Dimensional Case

We can also consider to solve $\nabla f(\mathbf{x}) = 0$.

By considering the Taylor's expansion at $\mathbf{x}^k$, we have

$$\nabla f(\mathbf{x}) \approx \nabla f(\mathbf{x}^k) + \nabla^2 f(\mathbf{x}^k)(\mathbf{x} - \mathbf{x}^k)$$

The solution to $\nabla f(\mathbf{x}) = 0$ is

$$\mathbf{x} = \mathbf{x}^k - (\nabla^2 f(\mathbf{x}^k))^{-1} \nabla f(\mathbf{x}^k)$$

which is also the Newton's step.

- Again, we assume $\nabla^2 f(\mathbf{x})$ is invertible in the search region

Connection to Descent Directions

Remember we call $\mathbf{d}$ a descent direction if $\nabla f(\mathbf{x})^T \mathbf{d} < 0$. That is, if we go a very small step in that direction, the objective value must be decreasing (due to Taylor expansion)

- In the gradient descent method, $\mathbf{d} = -\nabla f(\mathbf{x})$. Therefore,

$$\nabla f(\mathbf{x})^T \mathbf{d} = -||\nabla f(\mathbf{x})||^2 < 0$$

  Thus, the search direction in the gradient descent method is always a descent direction

Newton's Step

In Newton's method

$$\mathbf{d} = -(\nabla^2 f(\mathbf{x}))^{-1} \nabla f(\mathbf{x})$$

Then we have:

$$\nabla f(\mathbf{x})^T \mathbf{d} = -\nabla f(\mathbf{x})^T (\nabla^2 f(\mathbf{x}))^{-1} \nabla f(\mathbf{x})$$

If f is convex, then $\nabla^2 f(\mathbf{x})$ is positive semi-definite, therefore $\nabla f(\mathbf{x})^T \mathbf{d} \leq 0$. Furthermore, if $\nabla^2 f(\mathbf{x})$ is positive definite, then $\nabla f(\mathbf{x})^T \mathbf{d} < 0$

- In the following, we assume $f(\mathbf{x})$ is convex
- In this case, Newton's direction is always a descent direction

Step Length

As we said earlier, Newton's method may not converge unless the starting point is close.

One way to help convergence is to use a step size parameter α_k in

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$$

where $\mathbf{d}^k = -(\nabla^2 f(\mathbf{x}^k))^{-1} \nabla f(\mathbf{x}^k)$ is the Newton's step.

- One can use backtracking line search to determine α_k

Complete Procedure of Newton's Method

Start from a point $\mathbf{x}^0$, set tolerance $\epsilon > 0$ and $k = 0$.

1. Compute the Newton's step:

$$\mathbf{d}^k = -(\nabla^2 f(\mathbf{x}^k))^{-1} \nabla f(\mathbf{x}^k)$$

   If $||\nabla f(\mathbf{x}^k)|| < \epsilon$, then output $\mathbf{x}^k$, otherwise continue;
2. Choose step size α_k by backtracking line search
3. $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$, $k = k + 1$, go back to step 1

Convergence Results for Newton's Method

Theorem

Under some conditions and if $f(\mathbf{x})$ is convex, then the Newton's method converges in

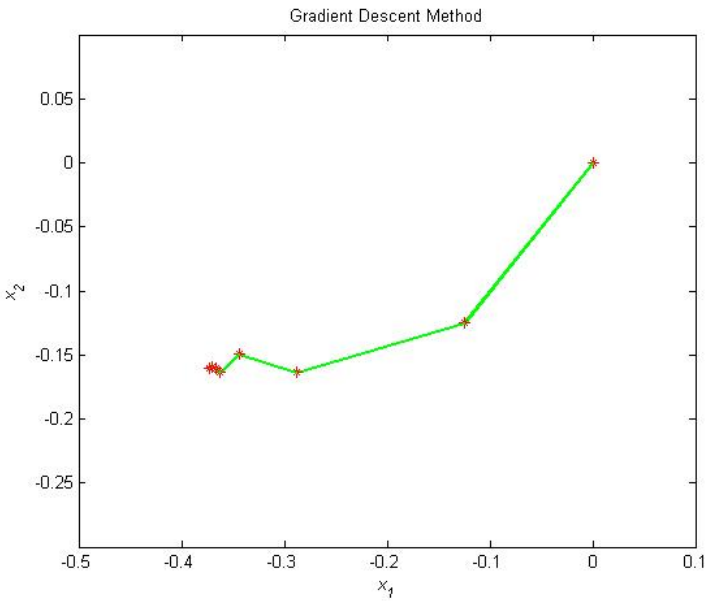
$$\frac{f(\mathbf{x}^0) - p^*}{\gamma} + 6$$

steps (to reach accuracy of 5×10^{-20}). Here p^* is the optimal value of the problem, γ is a coefficient depending on the problem instance (see Boyd's book p490 for detail).

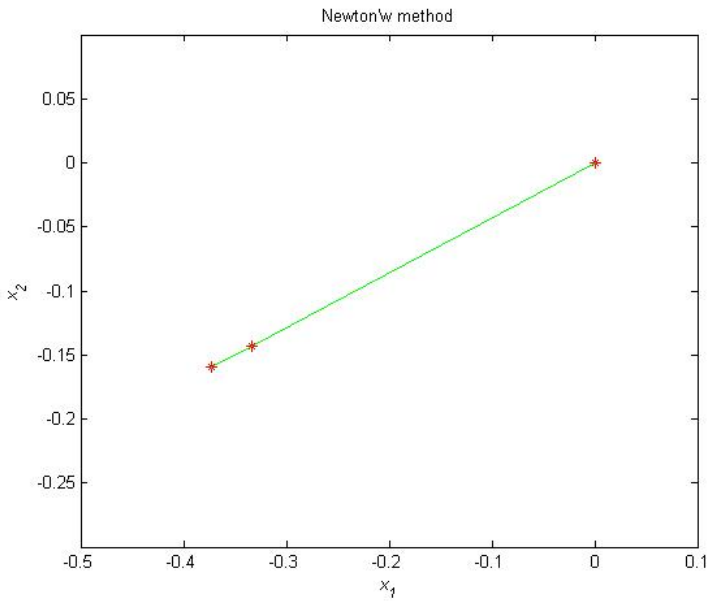
- If the original function is a quadratic function, then Newton's method will converge in just 1 step.

Example

$$\text{minimize} \quad e^{x_1+x_2} + x_1^2 + 3x_2^2 - x_1x_2$$



(a) Gradient Descent Method



(b) Newton's Method

What about the least squares problem?

- It will converge in 1 step.

Newton's Method is Fast, But...

It requires computing the second-order derivative in each iteration

- It might be computationally expensive to do so, especially if the second-order derivative doesn't have a closed-form (which is the case for many useful applications)
- It also requires a lot of space to store the Hessian matrix.
  - If it is an n-dimensional problem, then we need n^2 space, comparing to n space required for gradient descent method
- It requires much more matrix computation in each iteration

One partial solution (quasi-Newton method)

- We don't update $\nabla^2 f(\mathbf{x})$ in each step. Instead, we only update it once in a while
- It can save a lot of computation, although one needs to be careful

Gradient Descent Method vs Newton's Method

First, the two methods have the same framework

Start from a point $\mathbf{x}^0$, set tolerance $\epsilon > 0$ and $k = 0$.

1. If $||\nabla f(\mathbf{x}^k)|| < \epsilon$, then output $\mathbf{x}^k$, otherwise compute the search direction $\mathbf{d}^k$ and continue;
2. Choose step size α_k by backtracking line search
3. Set $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$, $k = k + 1$, go back to step 1

In gradient descent method

$$\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$$

In Newton's method

$$\mathbf{d}^k = -(\nabla^2 f(\mathbf{x}^k))^{-1} \nabla f(\mathbf{x}^k)$$

Tradeoff between Newton and Gradient Method

- Gradient method is fast in each iteration and needs less space, however, it takes much more iterations to converge
- Newton's method takes much less iterations to converge, however, each iteration is much more expensive in both computation and storage space

There are many numerical improvements that one can make but the main tradeoff remains

- Whether gradient method or Newton's method should be used depends largely on the problem at hand

Constrained Optimization

We consider the following constrained optimization problem:

$$\begin{aligned} \text{minimize}_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & h_i(\mathbf{x}) = 0, \quad \forall i \\ & g_j(\mathbf{x}) \leq 0, \quad \forall j \end{aligned}$$

In unconstrained method, the main idea is:

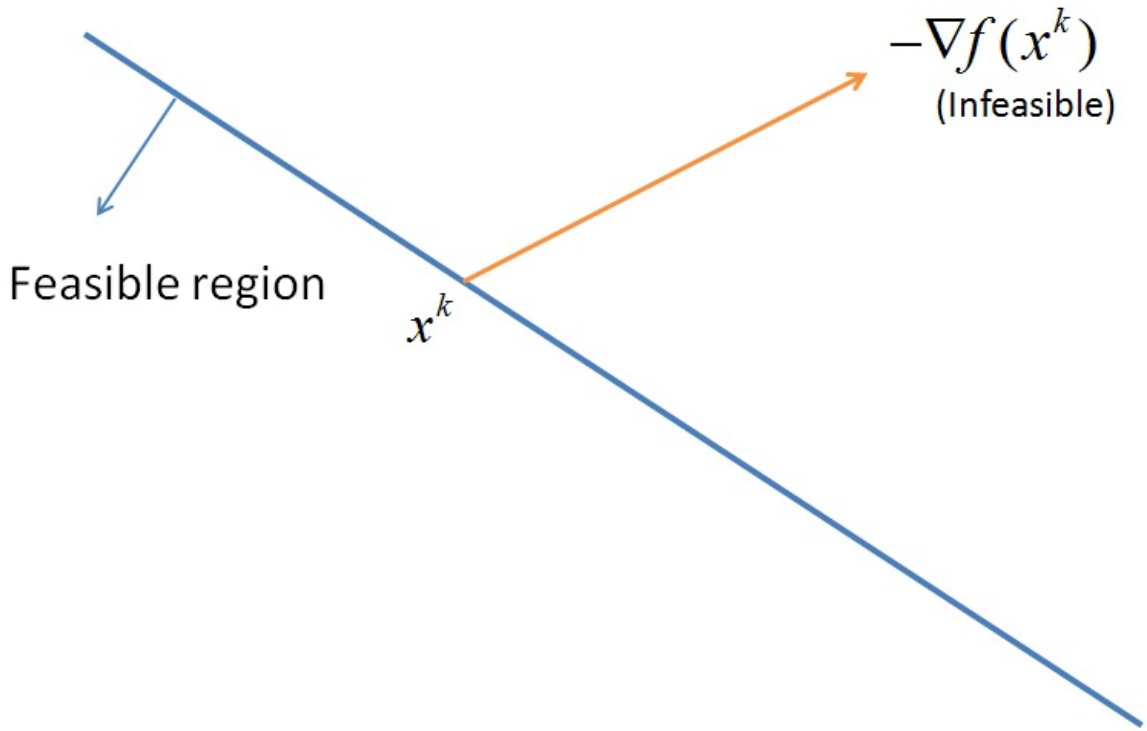
- At each $\mathbf{x}^k$, compute a descent direction, say $\mathbf{d}^k$
- Then find an appropriate step-size α_k and go to $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$
- Both gradient and Newton's methods are based on this idea

The problem when we have constraints

- $\mathbf{x}^{k+1}$ may become infeasible

In the following, we assume we have an initial feasible solution

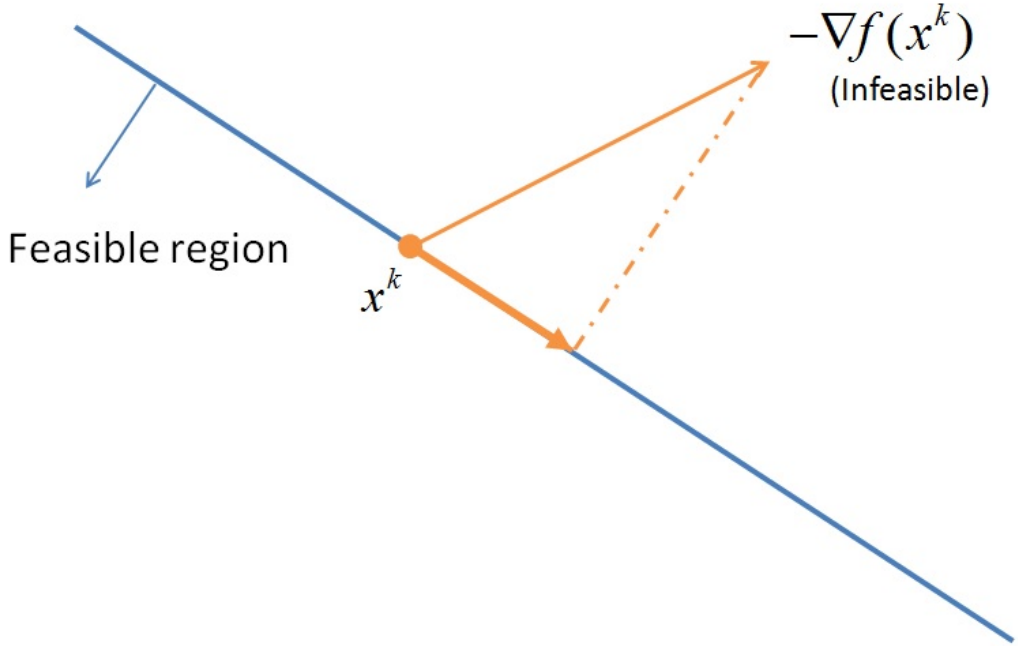
Illustration



- One of the solution to this problem is to use the gradient projection method

Main Idea of Gradient Projection Method

Project the descent direction onto the feasible set



We still use the general framework, however, the $\mathbf{d}^k$ chosen has to take into account feasibility — we will introduce the detail steps next week