# Multivariate Newton and the Gauss-Newton algorithm

Source: Ahmadi, ORF 363 slides.

## Overview

- In the previous lecture, we discussed steepest gradient descents, which use only first order information (first order derivatives)

- These algorithms achieved a linear rate of convergence

## Overview

- In the previous lecture, we discussed steepest gradient descents, which use only first order information (first order derivatives)

- These algorithms achieved a linear rate of convergence

- Today, we will discuss Newton’s method, which is a descent method with a specific choice of a descent direction

- We will see that the Newton’s method achieves quadratic rate of convergence

Newton’s method

The Armijo Rule

Levenberg-Marquardt Modification

Gauss-Newton method

## Newton’s method

Recall the general form of our descent methods:

$$x_{k+1} = x_k + \alpha_k d_k$$

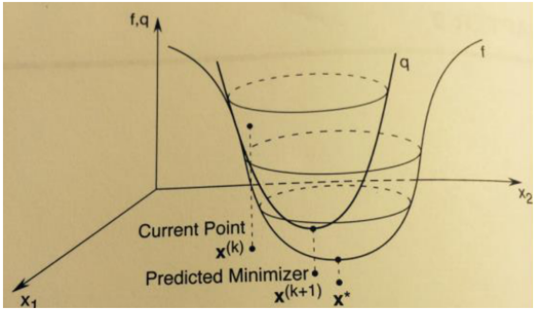
Let $\alpha_k = 1$ for now

The Newton’s method is the following choice of the descent direction:

$$d_k = -\left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k)$$

One motivation: minimizing a quadratic approximation of the function

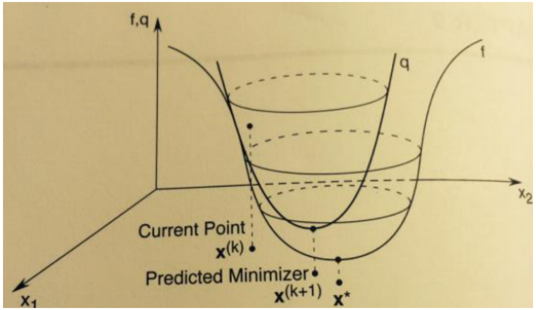
## Motivation



Suppose that we are at $x_k$. The Taylor's expansion of function $f$ up to the second order is:

$$f(x) \approx f(x_k) + \nabla f(x_k)^T (x - x_k) + \frac{1}{2}(x - x_k)^T \nabla^2 f(x_k)(x - x_k) = q(x)$$

## Motivation

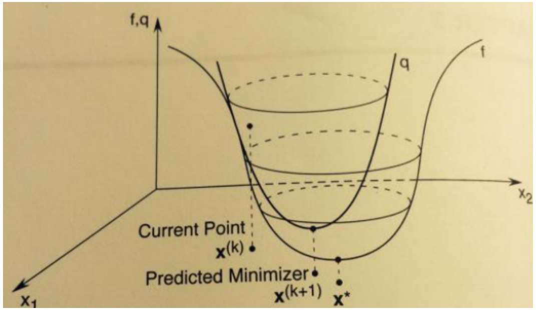


Suppose that we are at $x_k$. The Taylor's expansion of function $f$ up to the second order is:

$$f(x) \approx f(x_k) + \nabla f(x_k)^T(x - x_k) + \frac{1}{2}(x - x_k)^T \nabla^2 f(x_k)(x - x_k) = q(x)$$

$q$ is convex, since $\nabla^2 f(x_k) \succeq 0$. Thus stationary point $x^*$ of $q$ is the global optimum of $q$

## Motivation



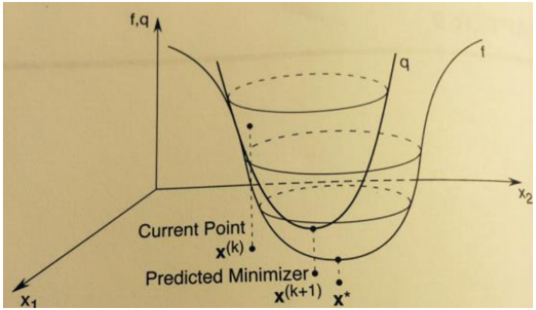
$$q(x) = f(x_k) + \nabla f(x_k)^T(x - x_k) + \frac{1}{2}(x - x_k)^T \nabla^2 f(x_k)(x - x_k)$$

$x^*$ is a stationary point of $q$

$$0 = \nabla q(x^*) = \nabla f(x_k) + \nabla^2 f(x_k)(x^* - x_k)$$

Solving for $x^*$,

## Motivation



$$x^* = x_k - \left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k)$$

Replace $x^*$ with the next point $x_{k+1}$

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k)$$

## Newton’s method for quadratic functions

- If $f(x) = \frac{1}{2}x^T Q x + b^T x + c$ with $Q \succ 0$, Newton’s method finds the global minimum in a single iteration
  - This is because $f(x) = q(x)$, and $x^*$ minimizes $q(x)$

## Newton's method for quadratic functions

- If $f(x) = \frac{1}{2}x^T Q x + b^T x + c$ with $Q \succ 0$, Newton's method finds the global minimum in a single iteration
    - This is because $f(x) = q(x)$, and $x^*$ minimizes $q(x)$
- When $f$ is not quadratic, we still have the following convergence result

## Convergence and rate of convergence

**Theorem.** Let $x^* \in \mathbb{R}^n$ such that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is invertible. Then, the iterations of the Newton's method starting from any point $x_0$ "near" $x^*$ converges to $x^*$ with quadratic convergence rate

## Convergence and rate of convergence

**Theorem.** Let $x^* \in \mathbb{R}^n$ such that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is invertible. Then, the iterations of the Newton's method starting from any point $x_0$ "near" $x^*$ converges to $x^*$ with quadratic convergence rate

**Interpretation:** there is a basin around stationary points such that once you are trapped in it, you converge to the stationary point very fast

## Convergence and rate of convergence

**Theorem.** Let $x^* \in \mathbb{R}^n$ such that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is invertible. Then, the iterations of the Newton's method starting from any point $x_0$ "near" $x^*$ converges to $x^*$ with quadratic convergence rate

**Interpretation:** there is a basin around stationary points such that once you are trapped in it, you converge to the stationary point very fast

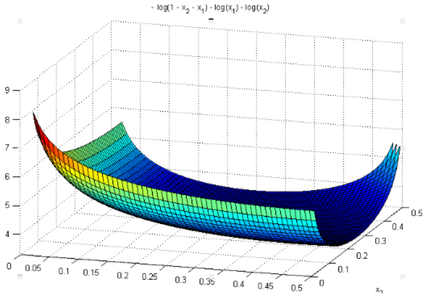
**Caution:** No guarantee that Nton iterations would converge if we started far away

## Example

$$f(x) = -\log(1 - x_1 - x_2) - \log x_1 - \log x_2$$

This function is convex (why?)

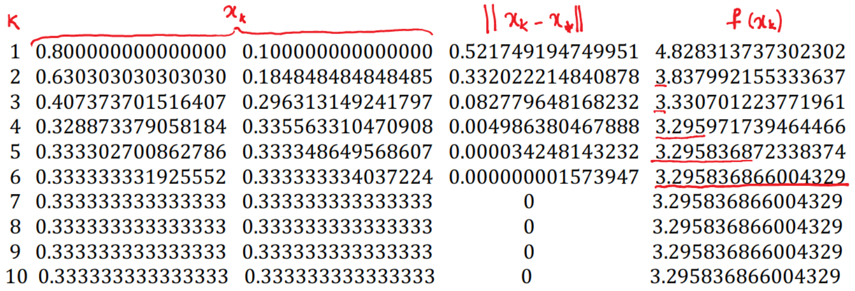
$$x^* = \left(\frac{1}{3}, \frac{1}{3}\right), \quad f(x^*) = 3.295836867$$



## Example

$$f(x) = -\log(1 - x_1 - x_2) - \log x_1 - \log x_2$$

$$x^* = \left(\frac{1}{3}, \frac{1}{3}\right), \quad f(x^*) = 3.295836867$$



| $K$ | $\overbrace{\qquad\qquad}^{x_k}$ | | $\lVert x_k - x_* \rVert$ | $f(x_k)$ |
|---|---|---|---|---|
| 1 | 0.800000000000000 | 0.100000000000000 | 0.521749194749951 | 4.828313737302302 |
| 2 | 0.630303030303030 | 0.184848484848485 | 0.332022214840878 | 3.837992155333637 |
| 3 | 0.407373701516407 | 0.296313149241797 | 0.082779648168232 | 3.330701223771961 |
| 4 | 0.328873379058184 | 0.335563310470908 | 0.004986380467888 | 3.295971739464466 |
| 5 | 0.333302700862786 | 0.333348649568607 | 0.000034248143232 | 3.295836872338374 |
| 6 | 0.333333331925552 | 0.333333334037224 | 0.000000001573947 | 3.295836866004329 |
| 7 | 0.333333333333333 | 0.333333333333333 | 0 | 3.295836866004329 |
| 8 | 0.333333333333333 | 0.333333333333333 | 0 | 3.295836866004329 |
| 9 | 0.333333333333333 | 0.333333333333333 | 0 | 3.295836866004329 |
| 10 | 0.333333333333333 | 0.333333333333333 | 0 | 3.295836866004329 |

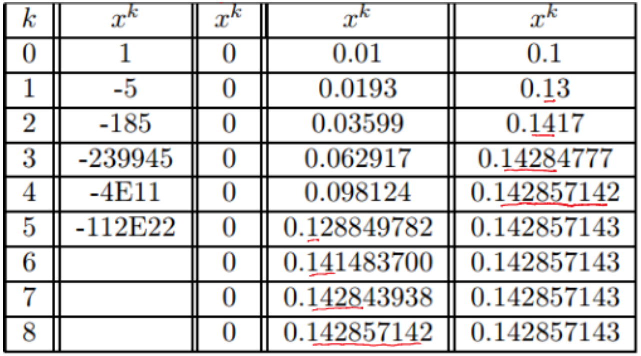
Number of correct significant digits doubles in each iteration

## Example

$$f(x) = 7x - \log x \qquad x^* = \frac{1}{7} = 0.1428571428$$

$$f'(x) = 1 - \frac{1}{x}, \quad f''(x) = \frac{1}{x^2}, \quad x_{k+1} =$$

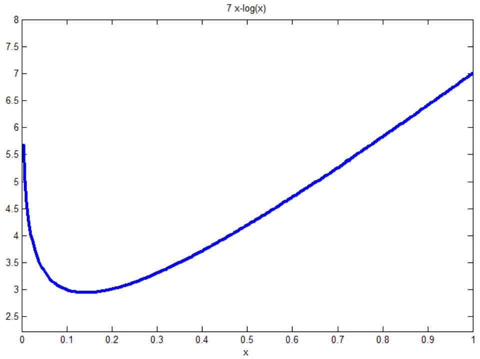
Four different initial conditions:



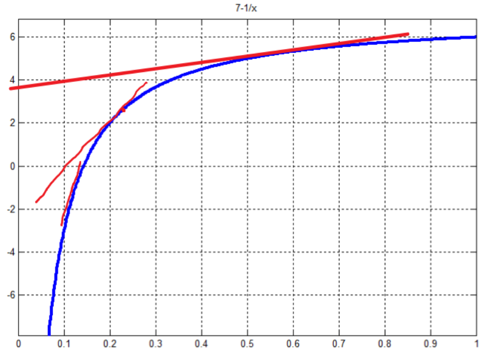
| $k$ | $x^k$ | $x^k$ | $x^k$ | $x^k$ |
|---|---|---|---|---|
| 0 | 1 | 0 | 0.01 | 0.1 |
| 1 | -5 | 0 | 0.0193 | 0.13 |
| 2 | -185 | 0 | 0.03599 | 0.1417 |
| 3 | -239945 | 0 | 0.062917 | 0.14284777 |
| 4 | -4E11 | 0 | 0.098124 | 0.142857142 |
| 5 | -112E22 | 0 | 0.128849782 | 0.142857143 |
| 6 |  | 0 | 0.141483700 | 0.142857143 |
| 7 |  | 0 | 0.142843938 | 0.142857143 |
| 8 |  | 0 | 0.142857142 | 0.142857143 |

## Good vs bad initial points

$$x_{k+1} = 2x_k - 7x_k^2$$



$f(x)$



$f'(x)$

The basin of attraction is $(0, \frac{2}{7})$ (why?)

## Newton’s method with a step size

$$x_{k+1} = x_k - \alpha_k \left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k)$$

- We saw many choices of the step size $\alpha_k$ in the previous lecture
- Let’s look at some popular ones

## The Armijo Rule

- This is an **inexact line search** method. It does not find the exact minimum along the line. But it guarantees sufficient decrease and it’s cheap

- Armijo rule requires two parameters: $\epsilon \in (0, 1), \delta > 1$

## The Armijo Rule

- This is an **inexact line search** method. It does not find the exact minimum along the line. But it guarantees sufficient decrease and it's cheap

- Armijo rule requires two parameters: $\epsilon \in (0, 1), \delta > 1$

Let $h(\alpha) = f(x_k + \alpha d_k)$. The goal is to minimize $h(\alpha)$

## The Armijo Rule

- This is an **inexact line search** method. It does not find the exact minimum along the line. But it guarantees sufficient decrease and it's cheap

- Armijo rule requires two parameters: $\epsilon \in (0,1), \delta > 1$

Let $h(\alpha) = f(x_k + \alpha d_k)$. The goal is to minimize $h(\alpha)$

Consider the following line:

$$\hat{h}(\alpha) = h(0) + \epsilon h'(0)\alpha$$

Note that $\hat{h}(0) = h(0) = f(x_k)$

## The Armijo Rule

- Armijo rule accepts a stepsize $\alpha$ if
  - $h(\bar{\alpha}) \leq \hat{h}(\bar{\alpha})$ (ensures sufficient decrease)
  - $h(\delta\bar{\alpha}) \geq \hat{h}\delta\bar{\alpha})$ (ensures step size is not too small)

## Armijo backtracking algorithm

- Start with some large initial step size $\alpha_0$
- At iteration $j$:
  - If $h(\alpha_j) \leq \hat{h}(\alpha_j)$, stop, and declare $\alpha_j$ as your step size
  - If $h\langle\alpha_j) > \hat{h}(\alpha_j)$, let $\alpha_{j+1} = \frac{1}{\delta}\alpha_j$

Newton’s method

The Armijo Rule

Levenberg-Marquardt Modification

Gauss-Newton method

## Levenberg-Marquardt Modification

**Issue:** Can't use Newton's method if $\nabla^2 f(x)$ is not invertible

## Levenberg-Marquardt Modification

**Issue:** Can’t use Newton’s method if $\nabla^2 f(x)$ is not invertible

**Idea:** Let’s make $\nabla^2 f(x)$ positive definite if it isn’t

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k) + \mu_k I\right)^{-1} \nabla f(x_k), \quad \mu_k \geq 0$$

# Levenberg-Marquardt Modification

**Issue:** Can't use Newton's method if $\nabla^2 f(x)$ is not invertible

**Idea:** Let's make $\nabla^2 f(x)$ positive definite if it isn't

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k) + \mu_k I\right)^{-1} \nabla f(x_k), \quad \mu_k \geq 0$$

**Lemma.** Let $A$ be an $n \times n$ matrix with eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$, and let $\mu \in \mathbb{R}$. Then the eigenvalues of $A + \mu I$ are $\lambda_1 + \mu, \lambda_2 + \mu, \ldots, \lambda_n + \mu$

## Remarks

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k) + \mu_k I\right)^{-1} \nabla f(x_k), \quad \mu_k \geq 0$$

- If $\mu_k$ is large enough, $\nabla^2 f(x_k) + \mu_k I$ will be invertible and by choosing a small enough step size $\alpha_k$ we can ensure descent

## Remarks

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k) + \mu_k I\right)^{-1} \nabla f(x_k), \quad \mu_k \geq 0$$

- If $\mu_k$ is large enough, $\nabla^2 f(x_k) + \mu_k I$ will be invertible and by choosing a small enough step size $\alpha_k$ we can ensure descent
- As $\mu_k \to 0$, we approach the regular Newton method

## Remarks

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k) + \mu_k I\right)^{-1} \nabla f(x_k), \quad \mu_k \geq 0$$

- If $\mu_k$ is large enough, $\nabla^2 f(x_k) + \mu_k I$ will be invertible and by choosing a small enough step size $\alpha_k$ we can ensure descent
- As $\mu_k \to 0$, we approach the regular Newton method
- As $\mu_k \to \infty$, we approach a pure gradient method with a small step size $\frac{\alpha_k}{\mu_k}$

## Remarks

$$x_{k+1} = x_k - \left(\nabla^2 f(x_k) + \mu_k I\right)^{-1} \nabla f(x_k), \quad \mu_k \geq 0$$

- If $\mu_k$ is large enough, $\nabla^2 f(x_k) + \mu_k I$ will be invertible and by choosing a small enough step size $\alpha_k$ we can ensure descent
- As $\mu_k \to 0$, we approach the regular Newton method
- As $\mu_k \to \infty$, we approach a pure gradient method with a small step size $\frac{\alpha_k}{\mu_k}$
- In practice, we can start with a small value of $\mu_k$ and increase it slowly until we observe descent: $f(x_{k+1}) < f(x_k)$

Newton’s method

The Armijo Rule

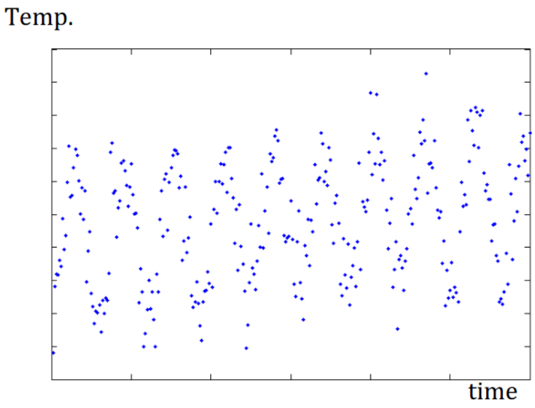
Levenberg-Marquardt Modification

Gauss-Newton method

## Gauss-Newton method for nonlinear least squares

Suppose you have observed the temperature in a town over the past several years. In view of seasonal effects and global warming, you postulate a model of the following form for the temperature at day $t$

$$T(t) = a \cdot \sin(\omega t + \phi) + bt$$

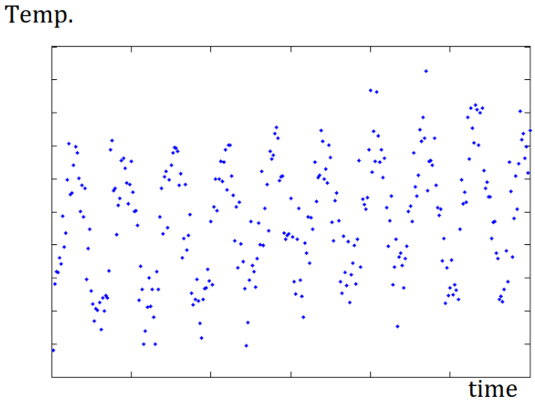


# Gauss-Newton method for nonlinear least squares

Suppose you have observed the temperature in a town over the past several years. In view of seasonal effects and global warming, you postulate a model of the following form for the temperature at day $t$

$$T(t) = a \cdot \sin(\omega t + \phi) + bt$$

The task is to find the parameters $a, \omega, \phi, b$ that best fit the data. Once this is done, we can use them to predict temperature at a future date



# Gauss-Newton method for nonlinear least squares

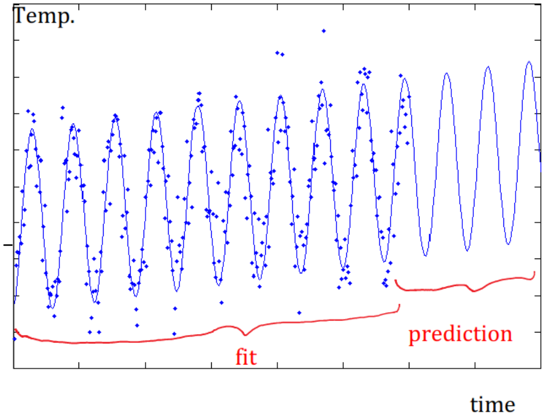
Denote the given data points by

$$(t_1, T_1), (t_2, T_2), \ldots, (t_m, T_m)$$

Goal is to minimize

$$f(a, \omega, \phi, b) = \sum_{i=1}^{m} (T_i - a \sim (\omega t_i + \phi) -$$

This is a **nonlinear least squares** problem



## Gauss-Newton method

More generally, we have a list of (possibly nonlinear) functions

$$g_1(x), g_2(x) \ldots, g_m(x), \quad g_i(x) : \mathbb{R}^n \to \mathbb{R}$$

and would like to minimize

$$f(x) = \frac{1}{2} \sum_{i=1}^{m} g_i^2(x)$$

The Gauss-Newton method is an approximation of Newton's method for minimizing this function

## Gauss-Newton method

$g_i(x) : \mathbb{R}^n \to \mathbb{R}, i = 1, \ldots, m$. We would like to minimize $f(x) = \frac{1}{2} \sum_{i=1}^{m} g_i^2(x)$

Let $g : \mathbb{R}^n \to \mathbb{R}^m$ be defined as $g = \begin{pmatrix} g_1 \\ \vdots \\ g_m \end{pmatrix}$

Let $J(x)$ be the Jacobian matrix of $g$, that is, $J_{ij}(x) = \frac{\partial g_i(x)}{\partial x_j}$. Then the Gauss-Newton iteration is

$$x_{k+1} = x_k - (J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$$

## Comments

$$x_{k+1} = x_k - (J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$$

- Unlike Newton, the Gauss-Newton method only uses first order informatio

## Comments

$$x_{k+1} = x_k - (J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$$

- Unlike Newton, the Gauss-Newton method only uses first order informatio
- The direction $-(J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$ is a descent direction, because $J(x_k)^T g(x_k)$ is the gradient of $f(x) = \frac{1}{2} \sum g_i^2(x)$ and $(J(x_k)^T J(x_k))^{-1}$ is positive definite (why?)

## Comments

$$x_{k+1} = x_k - (J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$$

- Unlike Newton, the Gauss-Newton method only uses first order informatio
- The direction $-(J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$ is a descent direction, because $J(x_k)^T g(x_k)$ is the gradient of $f(x) = \frac{1}{2} \sum g_i^2(x)$ and $(J(x_k)^T J(x_k))^{-1}$ is positive definite (why?)
- If $(J(x_k)^T J(x_k))^{-1}$ is not invertible then we can apply the Levenberg-Marquardt modification

## Comments

$$x_{k+1} = x_k - (J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$$

- If you were to write down the Newton iteration for minimizing you would get:

$$x_{k+1} = x_k - \left( J(x_k)^T J(x_k) + \sum_{i=1}^{m} \nabla^2 g_i^2(x_k) \right)^{-1} J(x_k)^T g(x_k)$$

## Comments

$$x_{k+1} = x_k - (J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$$

- If you were to write down the Newton iteration for minimizing you would get:

$$x_{k+1} = x_k - \left( J(x_k)^T J(x_k) + \sum_{i=1}^{m} \nabla^2 g_i^2(x_k) \right)^{-1} J(x_k)^T g(x_k)$$

- In Gauss-Newton, we ignore the term $\sum_{i=1}^{m} \nabla^2 g_i^2(x_k)$. This is a good approximation when $g_i$’s are close to linear function or when $g_i$’s are small

## Comments

$$x_{k+1} = x_k - (J(x_k)^T J(x_k))^{-1} J(x_k)^T g(x_k)$$

- If you were to write down the Newton iteration for minimizing you would get:

$$x_{k+1} = x_k - \left( J(x_k)^T J(x_k) + \sum_{i=1}^{m} \nabla^2 g_i^2(x_k) \right)^{-1} J(x_k)^T g(x_k)$$

- In Gauss-Newton, we ignore the term $\sum_{i=1}^{m} \nabla^2 g_i^2(x_k)$. This is a good approximation when $g_i$'s are close to linear function or when $g_i$'s are small

- If $g_i$'s are all linear functions, then we have a least squares problem

## Derivation of Gauss-Newton

Optimize $f(x) = \frac{1}{2}\sum_{i=1}^{m}(g_i(x))^2, \qquad g = \begin{pmatrix} g_1 \\ \vdots \\ g_m \end{pmatrix}$

Let $J(x)$ be the Jacobian matrix of $g$; i.e., $J_{ij}(x) = \frac{\partial g_i(x)}{\partial x_j}$

1. Replace $g(x)$ with its first order approximation $\tilde{g}(x, x_k)$ near the current iterate $x_k$ iterate

## Derivation of Gauss-Newton

Optimize $f(x) = \frac{1}{2}\sum_{i=1}^{m}(g_i(x))^2, \qquad g = \begin{pmatrix} g_1 \\ \vdots \\ g_m \end{pmatrix}$

Let $J(x)$ be the Jacobian matrix of $g$; i.e., $J_{ij}(x) = \frac{\partial g_i(x)}{\partial x_j}$

2. Instead of minimizing $\frac{1}{2}\|g(x)\|^2$, we minimize $\frac{1}{2}\|\tilde{g}(x, x_k)\|^2$. This is a quadratic function