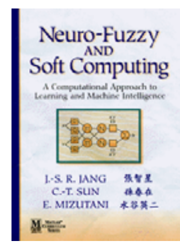
Chapter 6: Derivative-Based Optimization

- Introduction (6.1)
- Descent Methods (6.2)
- The Method of Steepest Descent (6.3)
- Newton's Methods (NM) (6.4)
- Step Size Determination (6.5)
- Nonlinear Least-Squares Problems (6.8)



Jyh-Shing Roger Jang et al., *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, First Edition, Prentice Hall, 1997

Introduction (6.1)

- Goal: Solving minimization nonlinear problems through derivative information
  - We cover:
    - Gradient based optimization techniques
    - Steepest descent methods
    - Newton Methods
    - Conjugate gradient methods
    - Nonlinear least-squares problems
  - They are used in:
    - Optimization of nonlinear neuro-fuzzy models
    - Neural network learning
    - Regression analysis in nonlinear models

Descent methods (6.2)

- Goal: Determine a point $\theta = \overset{*}{\theta} = \left[\overset{*}{\theta}_1, \overset{*}{\theta}_2, ..., \overset{*}{\theta}_n\right]^T$ such that

  $f(\theta_1, \theta_2, \ldots, \theta_n)$ is minimum on $\theta = \overset{*}{\theta}$.

- We are looking for a local & not necessarily a global minimum $\overset{*}{\theta}$

- Let $f(\theta_1, \theta_2, \ldots, \theta_n) = E(\theta_1, \theta_2, \ldots, \theta_n)$, the search of this minimum is performed through a certain direction d starting from an initial value $\theta = \theta_0$ (iterative scheme!)

Descent Methods (6.2) (cont.)

$$\theta_{next} = \theta_{now} + \eta\, d$$

($\eta > 0$ is a step size regulating the search in the direction d)

$$\theta_{k+1} = \theta_k + \eta_k d_k \quad (k = 1, 2, \ldots)$$

The series $\{\theta_k\}_{k=1,2,...,}$ should converge to a local minimum $\overset{*}{\theta}$

- We first need to determine the next direction d & then compute the step size η

- $\eta_k d_k$ is called the k-th step, whereas η_k is the k-th step size

- We should have $E(\theta_{next}) = E(\theta_{now} + \eta\, d) < E(\theta_{now})$

- The principal differences between various descent algorithms lie in the first procedure for determining successive directions

Descent Methods (6.2) (cont.)

- Once d is determined, $\overset{*}{\eta}$ is computed as:

$$\overset{*}{\eta} = \arg\min_{\eta>0} \varnothing(\eta)$$

$$\text{where}: \quad \varnothing(\eta) = E(\theta_{now} + \eta d)$$

- Gradient-based methods
  - Definition: The gradient of a differentiable function E: $IR^n \rightarrow IR$ at θ is the vector of first derivatives of E, denoted as g. That is:

$$g(\theta) = \nabla E(\theta) \overset{def}{=} \left[\frac{\partial E(\theta)}{\partial \theta_1}, \frac{\partial E(\theta)}{\partial \theta_2}, \ldots, \frac{\partial E(\theta)}{\partial \theta_n} \right]^T$$

Descent Methods (6.2) (cont.)

- Based on a given gradient, downhill directions adhere to the following condition for feasible descent directions:

$$\varnothing'(0) = \frac{dE(\theta_{now} + \eta d)}{d\eta}\bigg|_{\eta=0} = g^T d = \|g^T\| \|d\| \cos(\xi(\theta_{now})) < 0$$

Where ξ is the angle between g and d and $\xi(\theta_{now})$ is the angle between g_{now} and d at point θ_{now}

Descent models (6.2) (cont.)

The previous equation is justified by Taylor series expansion:

$$E(\theta_{now} + \eta d) = E(\theta_{now}) + \eta g^T d + 0(\eta^2)$$

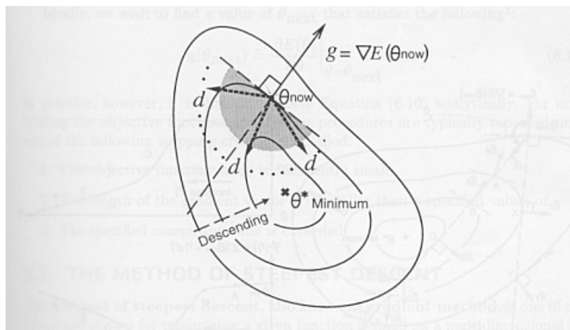


Figure 6.1. *Feasible descent directions. Directions from the starting point θ_{now} in the shaded area are possible descent vector candidates. When $\mathbf{d} = -\mathbf{g}$, $\mathbf{d}$ is the steepest descent direction at a local point θ_{now}.*

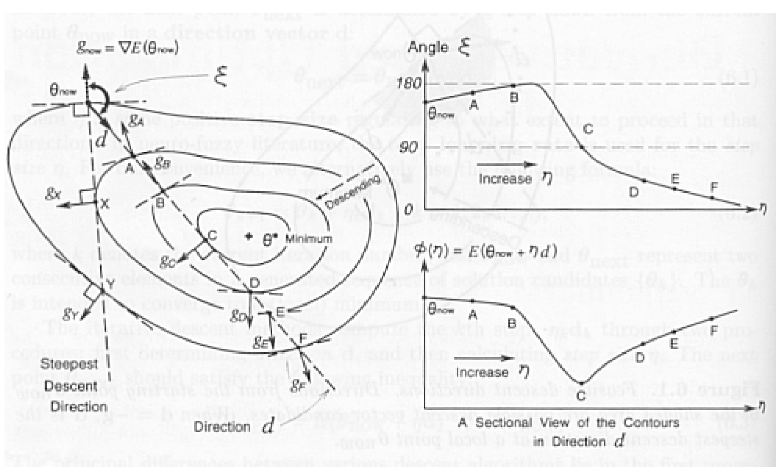


Figure 6.2. *Angle ξ between gradient directions $\mathbf{g}$ and a descent direction $\mathbf{d}$, which is determined by a certain algorithm at the current point θ_{now}. Let N be the set of all possible next points; $N \supset \{A, B, C, D, E, F, X, Y\}$. In the one-way downhill direction $\mathbf{d}$, the next point θ_{next} may be one of six points—A, B, C, D, E, or F—or be in the vicinity of them, depending on step sizes. By comparison, in the steepest descent direction, θ_{next}, may be either X or Y, or close to them.*

Descent Methods (6.2) (cont.)

- A class of gradient-based descent methods has the following form in which feasible descent directions can be found by gradient deflection
- Gradient deflection consists of multiplying the gradient g by a positive definite matrix (pdm) G

  $d = - Gg \Rightarrow g^T d = - g^T G g < 0$ (feasible descent direction)
- The gradient-based method is described therefore by:

  $$\theta_{next} = \theta_{now} - \eta G g \quad (\eta > 0,\ G \text{ pdm}) \qquad (*)$$

Descent Methods (6.2) (cont.)

- <u>Theoretically</u>, we wish to determine a value θ_{next} such as:

  $$\mathbf{g}(\theta_{\mathbf{next}}) = \frac{\partial \mathbf{E}(\theta)}{\partial \theta}\Big|_{\theta=\theta_{\mathbf{next}}} = \mathbf{0}$$

  but this is difficult to solve!!
- <u>But practically</u>, we stop the algorithm if:
  - The objective function value is sufficiently small
  - The length of the gradient vector g is smaller than a threshold
  - The computation time is exceeded

The method of Steepest Descent (6.3)

- Despite its slow convergence, this method is the most frequently used nonlinear optimization technique due to its simplicity
- If $G = I_d$ (identity matrix) then equation (*) expresses the steepest descent scheme:

$$\theta_{next} = \theta_{now} - \eta g$$

- If Cos $\xi = -1$ (meaning that d points to the same direction of vector – g) then the objective function E can be decreased locally by the biggest amount at point θ_{now}

The method of Steepest Descent (6.3) (cont.)

- Therefore, the negative gradient direction (-g) points to the locally steepest downhill direction
- This direction may not be a shortcut to reach the minimum point θ^*
- However, if the steepest descent uses the line minimization technique (min $\varnothing(\eta)$) then $\varnothing'(\eta) = 0$

$$\varnothing'(\eta) = \frac{\mathbf{dE}(\theta_{now} - \eta \mathbf{g}_{now})}{\mathbf{d}\eta} = \nabla^T \mathbf{E}(\theta_{now} - \eta \mathbf{g}_{now})\mathbf{g}_{now}$$

$$= \mathbf{g}_{next}^T - \mathbf{g}_{now} = \mathbf{0}$$

$\Rightarrow g_{next}$ is orthogonal to the current gradient vector g_{now} (see figure 6.2; pt X)

The method of Steepest Descent (6.3) (cont.)

- If the contours of the objective function E form hyperspheres (or circles in a 2 dimensional space), the steepest descent methods leads to the minimum in a single step. Otherwise the method does not lead to the minimum point

Newton's Methods (NM) (6.4)

- Classical NM
  - Principle: The descent direction d is determined by using the second derivatives of the objective function E if available
- If the starting position θ_{now} is sufficient close to a local minimum, the objective function E can be approximated by a quadratic form:

$$E(\theta) \cong E(\theta_{now}) + g^T(\theta - \theta_{now}) + \frac{1}{2}(\theta - \theta_{now})^T H(\theta - \theta_{now})$$

$$\text{where} \quad H = \nabla^2 E(\theta) = \left(\frac{\partial^2 E}{\partial^2 \theta}\right)$$

Newton's Methods (NM) (6.4) (cont.)

- Since the equation defines a quadratic function $E(\theta)$ in the θ_{now} neighborhood $\Rightarrow$ its minimum $\hat{\theta}$ can be determined by differenting & setting to 0. Which gives:

  $$0 = g + H(\hat{\theta} - \theta_{now})$$

  Equivalent to: $\hat{\theta} = \theta_{now} - H^{-1}g$

- It is a gradient-based method for $\eta = 1$ and $G = H^{-1}$

Newton's Methods (NM) (6.4) (cont.)

- Only when the minimum point $\hat{\theta}$ of the approximated quadratic function is chosen as the next point θ_{next}, we have the so-called NM or the Newton-Raphson method

  $$\hat{\theta} = \theta_{now} - H^{-1}g$$

- If H is positive definite and $E(\theta)$ is quadratic then the NM directly reaches a local minimum in the single Newton step (single $- H^{-1}g$)

- If $E(\theta)$ is not quadratic, then the minimum may nor be reached in a single step & NM should be iteratively repeated

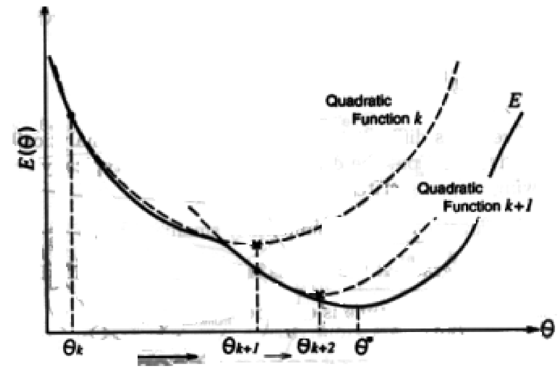


Figure 6.3. *Newton's (or Newton-Raphson) method for minimizing a general obj tive function E, which is approximated locally as a quadratic form; this approxim function is minimized exactly.*

Step Size Determination (6.5)

- Formula of a class of gradient-based descent methods:

$$\theta_{next} = \theta_{now} + \eta d = \theta_{now} - \eta G g$$

- This formula entails effectively determining the step size η

- $\varnothing'(\eta) = 0$ with $\varnothing(\eta) = E(\theta_{now} + \eta d)$ is often impossible to solve

Step Size Determination (6.5) (cont.)

- Initial Bracketing
  - We assume that the search area (or specified interval) contains a single relative minimum: E is unimodal over the closed interval
  - Determining the initial interval in which a relative minimum must lie is of critical importance
    - A scheme, by function evaluation for finding three points to satisfy:
      $E(\theta k-1) > E(\theta k) < E(\theta k+1)$; $\theta k-1 < \theta k < \theta k+1$
    - A scheme, by taking the first derivative, for finding two points to satisfy:
    - $E'(\theta k) < 0$, $E'(\theta k+1) > 0$, $\theta k < \theta k+1$

- **Algorithm for scheme 1:**
  An initial bracketing for searching three points θ_1, θ_2 and θ_3

1) Given a starting point θ_0 and $h \in IR$, let θ_1 be θ_0 +h.
   Evaluate $E(\theta_1)$
   if $E(\theta_0) \geq E(\theta_1)$, i $\leftarrow$1
   (i.e., go downhill) go to (2)
   otherwise h $\leftarrow$ -h (i.e., set backward direction)
   $E(\theta_{-1}) \leftarrow E(\theta_1)$
   $\theta_1 \leftarrow \theta_0 + h$
   $i \leftarrow 0$
   go to (3)

2) Set the next point by; $h \leftarrow 2h$, $\theta_{i+1} \leftarrow \theta_i + h$
3) Evaluate $E(\theta_{i+1})$
   if $E(\theta_i) \geq E(\theta_{i+1})$; $i \leftarrow i + 1$
   (i.e., still go downhill) go to (2)
   Otherwise, Arrange θ_{i-1}, θ_i and θ_{i+1} in the decreasing order
   Then, we obtain the three points: $(\theta_1,\theta_2,\theta_3)$
   Stop.

Step Size Determination (6.5) (cont.)

- Line searches
  - The process of determining η^* that minimizes a one-dimensional function $\varnothing(\eta)$ is achieved by searching on the line for the minimum
  - Line search algorithms usually include two components: sectioning (or bracketing), and polynomial interpolation
    - Newton's method
      When $\varnothing(\eta k)$, $\varnothing'(\eta k)$, and $\varnothing''(\eta k)$ are available, the classical Newton method (defined by $\hat{\theta} = \theta_{now} - H_g^{-1}$) can be applied to solving the equation $\varnothing'(\eta k) = 0$:

$$\eta_{k+1} = \eta_k - \frac{\varnothing'(\eta_k)}{\varnothing''(\eta_k)} \qquad (*)$$

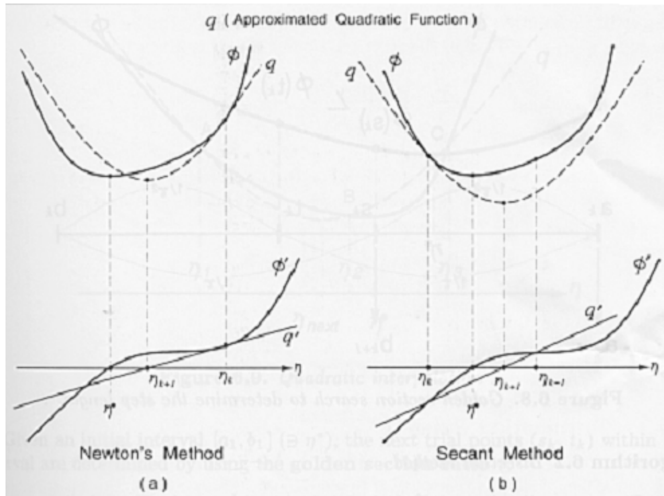
Step Size Determination (6.5) (cont.)

- Secant method
  If we use both ηk and ηk-1 to approximate the second derivative in equation (*), and if the first derivatives alone are available then we have an estimated ηk+1 defined as:

$$\eta_{k+1} = \eta_k - \frac{\varnothing'(\eta_k)}{\dfrac{\varnothing'(\eta_k) - \varnothing'(\eta_{k-1})}{\eta_k - \eta_{k-1}}}$$

this method is called the secant method.

Both the Newton's and the secant method are illustrated in the following figure.



Newton's method and secant method to determinethe step size

Step Size Determination (6.5) (cont.)

- Sectioning methods
  - It starts with an interval [a1, b1] in which the minimum η^* must lie, and then reduces the length of the interval at each iteration by evaluating the value of ∅ at a certain number of points
  - The two endpoints a1 and b1 can be found by the initial bracketing described previously
  - The bisection method is one of the simplest sectioning method for solving $\varnothing'(\eta^*) = 0$, if first derivatives are available!

Let $\varnothing'(\eta) = \varphi(\eta)$ then the algorithm is:

Algorithm [bisection method]

(1) Given $\varepsilon \in IR^+$ and an initial interval with 2 endpoints a_1 and a_2 such that: $a_1 < a_2$ and $\varphi(a_1)\varphi(a_2) < 0$ then set:

$$\eta_{left} \leftarrow a_1$$
$$\eta_{right} \leftarrow a_2$$

(2) Compute the midpoint η_{mid}; $\eta_{mid} \leftarrow (\eta_{right} + \eta_{left}) / 2$
if $\varphi(\eta_{right})\ \varphi(\eta_{mid}) < 0$, $\eta_{left} \leftarrow \eta_{mid}$
Otherwise $\eta_{right} \leftarrow \eta_{mid}$

(3) Check if $|\eta_{left} - \eta_{right}| < \varepsilon$. If it is true then terminate the algorithm, otherwise go to (2)

Step Size Determination (6.5) (cont.)

- Golden section search method

This method does not require $\varnothing$ to be differentiable. Given an initial interval [a1,b1] that contains $\overset{*}{\eta}$, the next trial points (sk,tk) within the interval are determined by using the golden section ratio τ:

$$s_k = b_k - \frac{1}{\tau}(b_k - a_k) = b_k + \frac{\tau - 1}{\tau}(b_k - a_k)$$

$$t_k = a_k + \frac{1}{\tau}(b_k - a_k)$$

$$\text{where} \quad \tau = \frac{1+\sqrt{5}}{2} \cong 1.618$$

Step Size Determination (6.5) (cont.)

This procedure guarantees the following:

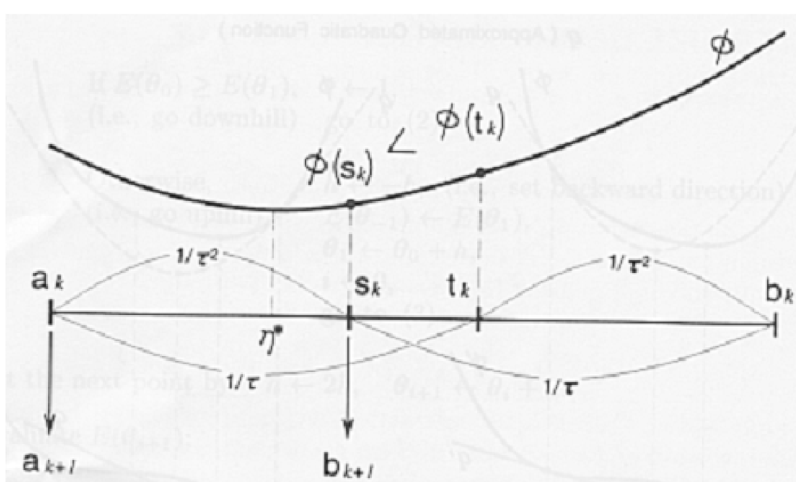
ak < sk < tk < bk

The algorithm generates a sequence of two endpoints ak and bk, according to:

If ∅(sk) > ∅(tk), ak+1 = sk, bk+1 = bk

Otherwise ak+1 = ak, bk+1 = tk

The minimum point η^* is bracketed to an interval just 2/3 times the length of the preceding interval



Golden section search to determine the step length

Step Size Determination (6.5) (cont.)

- Line searches (cont.)
  - Polynomial interpolation
    - This method is based on curve-fitting procedures
    - A quadratic interpolation is the method that is very often used in practice
    - It constructs a smooth quadratic curve q that passes through three points $(\eta_1, \varnothing_1)$, $(\eta_2, \varnothing_2)$ and $(\eta_3, \varnothing_3)$:

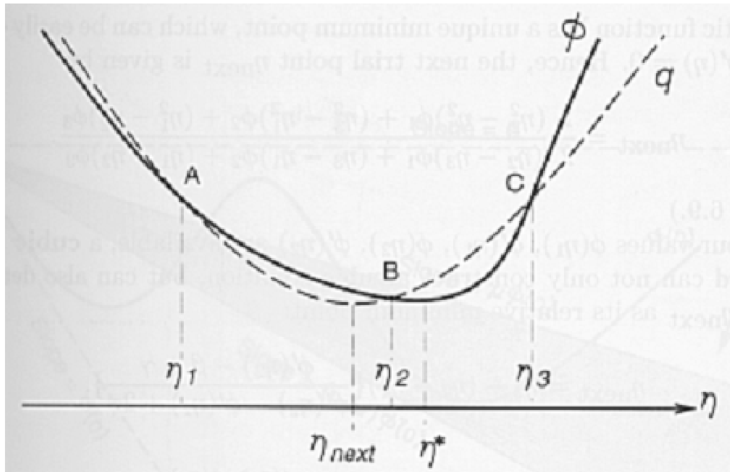
$$q(\eta) = \sum_{i=1}^{3} \varnothing_i \frac{\prod_{j \neq i} (\eta - \eta_j)}{\prod_{j \neq i} (\eta_i - \eta_j)}$$

where $\varnothing_i = \varnothing(\eta_i)$, i = 1, 2, 3

Step Size Determination (6.5) (cont.)

- Polynomial interpolation (cont.)
  - Condition for obtaining a unique minimum point is: $q'(\eta) = 0$, therefore the next point η_{next} is:

$$\eta_{next} = \frac{1}{2} * \frac{(\eta_2^2 - \eta_3^2)\varnothing_1 + (\eta_3^2 - \eta_1^2)\varnothing_2 + (\eta_1^2 - \eta_2^2)\varnothing_3}{(\eta_2 - \eta_3)\varnothing_1 + (\eta_3 - \eta_1)\varnothing_2 + (\eta_1 - \eta_2)\varnothing_3}$$



Quadratic Interpolation

Step Size Determination (6.5) (cont.)

- Termination rules
  - Line search methods do not provide the exact minimum point of the function ∅
  - We need a termination rule that accelerate the entire minimization process without affecting too much precision

Step Size Determination (6.5) (cont.)

- Termination rules (cont.)
  - The Goldstein Test
    - This method is based on two definitions:
      - A value of η is not too large if with a given μ $(0 < \mu < ½)$,
        $$\varnothing(\eta) \le \varnothing(0) + \mu\, \varnothing'(0)\eta$$
      - A value of η is considered to be not too small if:
        $$\varnothing(\eta) > \varnothing(0) + (1 - \mu)\, \varnothing'(\eta)$$

Step Size Determination (6.5) (cont.)

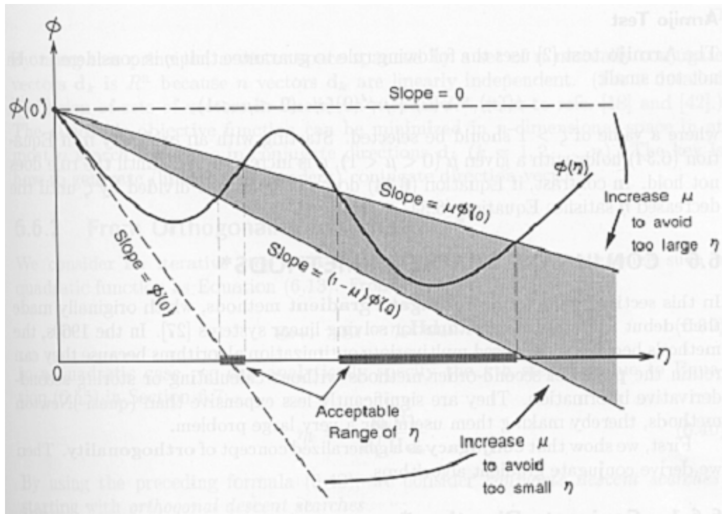
- Goldstein test (cont.)
  - From the two precedent inequalities, we obtain:

$(1 - \mu)\, \varnothing'(0)\eta \le \varnothing(\eta) - \varnothing(0) = E(\theta next) – E(\theta now) \le \mu\, \varnothing'(0)\eta$

which can be written as:

$$0 < \mu \le \frac{E(\theta_{next}) - E(\theta_{now})}{\eta g d} \le 1 - \mu < 1$$

where $\varnothing'(0) = gTd < 0$ (Taylor series)



Goldstein Test

Nonlinear Least-Squares Problems (6.8)

- Goal: Optimize a model by minimizing a squared error measure between desired outputs & the model's output

$y = f(x, \theta)$

Given a set of m training data pairs (xp; tp), (p = 1, …, m), we can write:

$$\mathbf{E}(\theta) = \sum_{p=1}^{m} (t_p - y_p)^2 = \sum_{p=1}^{m} (t_p - f(x_p, \theta))^2$$

$$= \sum_{p=1}^{m} r_p(\theta)^2 = r^T(\theta).r(\theta)$$

Nonlinear Least-Squares Problems (6.8) (cont.)

- The gradient is expressed as:

$$\mathbf{g} = \mathbf{g}(\mathbf{0}) = \frac{\partial \mathbf{E}(\theta)}{\partial \theta} = \mathbf{2}\sum_{\mathbf{p=1}}^{\mathbf{m}} \mathbf{r_p}(\theta) \frac{\partial \mathbf{r_p}(\theta)}{\partial \theta} = \mathbf{2J^T.r}$$

where J is the Jacobian matrix of r.

$$\left((\mathbf{r}, \theta) \xrightarrow{\varphi} (\mathbf{r}\cos\theta, \mathbf{r}\sin\theta) \quad \mathbf{J}_\varphi = \mathbf{r} \right)$$

Since $r_p(\theta) = t_p - f(x_p, \theta)$, this implies that the pth row of J is:

$$-\nabla_\theta^T \mathbf{f}(\mathbf{x_p}, \theta)$$

Nonlinear Least-Squares Problems (6.8) (cont.)

- Gauss-Newton Method
  - Known also as the linearization method
  - Use Taylor series expansion to obtain a linear model that approximates the original nonlinear model
  - Use linear least-squares optimization of chapter 5 to obtain the model parameters

Nonlinear Least-Squares Problems (6.8) (cont.)

- Gauss-Newton Method (cont.)
  - The parameters $\theta^T = (\theta_1, \theta_2, \ldots, \theta_n, \ldots)$ will be computed iterativelly
  - Taylor expansion of $y = f(x, \theta)$ around $\theta = \theta_{now}$

$$\mathbf{y} = \mathbf{f}(\mathbf{x}, \theta_{\mathbf{now}}) + \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \left(\frac{\partial \mathbf{f}(\mathbf{x}, \theta)}{\partial \theta_{\mathbf{i}}} \bigg|_{\theta=\theta_{\mathbf{now}}} \right) \left(\theta_{\mathbf{i}} - \theta_{\mathbf{i,now}}\right)$$

Nonlinear Least-Squares Problems (6.8) (cont.)

- Gauss-Newton Method (cont.)
  - $y - f(x, \theta_{now})$ is linear with respect to $\theta_i - \theta_{i,now}$ since the partial derivatives are constant

$$\mathbf{E}(\theta) = \left\| \mathbf{t} - \mathbf{f}(\mathbf{x}, \theta_{\mathbf{now}}) - \frac{\partial \mathbf{f}(\mathbf{x}, \theta_{\mathbf{now}})}{\partial \theta}(\theta - \theta_{\mathbf{now}}) \right\|^2$$

$$= \left\| \mathbf{r} + \mathbf{J}^{\mathbf{T}}(\theta - \theta_{\mathbf{now}}) \right\|^2 = \left\| \mathbf{r} + \mathbf{J}^{\mathbf{T}}\mathbf{S} \right\|^2$$

where $S = \theta - \theta_{now}$

Nonlinear Least-Squares Problems (6.8) (cont.)

- Gauss-Newton Method (cont.)
  - The next point θ_{next} is obtained by:

$$\frac{\partial \mathbf{E}(\theta)}{\partial \theta}\bigg|_{\theta=\theta_{\mathbf{next}}} = \mathbf{J}^{\mathbf{T}}\left\{\mathbf{r} + \mathbf{J}(\theta_{\mathbf{next}} - \theta_{\mathbf{now}})\right\} = \mathbf{0}$$

  - Therefore, the following Gauss-Newton formula is expressed as:

$$\theta_{next} = \theta_{now} - (J^T J)^{-1} J^T r = \theta_{now} - \tfrac{1}{2} (J^T J)^{-1} g$$

(since $g = 2J^T r$)