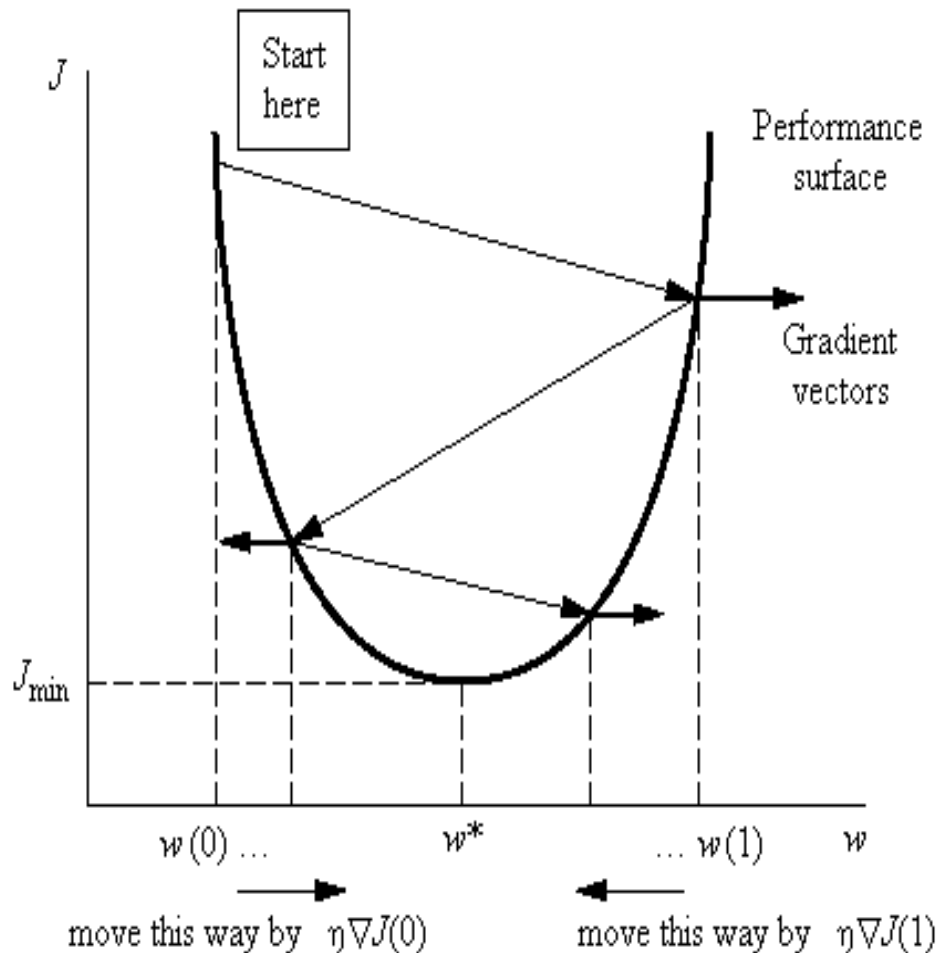


Derivative-Based Optimization

Goal: Solving minimization nonlinear problems through derivative information

- ✓ Descent Methods
- ✓ The Method of Steepest Descent
- ✓ Newton's Methods (NM)
- ✓ 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

Derivative Based Optimization



The search using the gradient information

The gradient of the performance surface is a vector (with the dimension of w) that always points toward the direction of maximum E change and with a magnitude equal to the slope of the tangent of the performance surface.

Visualize the performance surface as a hillside, each point on the hill will have a gradient arrow that points in the direction of steepest ascent at that point, with larger magnitudes for steeper slopes.

$$w(k+1) = w(k) - \eta \nabla J(k)$$

Descent Methods

- Goal: Determine a point such that

$$\theta = \theta^* = \begin{bmatrix} \theta_1^*, \theta_2^*, \dots, \theta_n^* \end{bmatrix}^T$$

$f(\theta_1, \theta_2, \dots, \theta_n)$ is minimum on

$$\theta = \theta^*.$$

- We are looking for a local & not necessarily a global minimum θ^*

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

- Minimizing a real valued objective function $E(\theta)$ defined on an n dimensional input space
 $\theta = [\theta_1, \theta_2, \dots, \theta_n]^T$
- Finding (possibly local) a minimum point
 $\theta = \theta^*$ that minimizes $E(\theta)$
- θ_{k+1} is determined by step down point in a d direction
 $\theta_{k+1} = \theta_k + \eta_k \mathbf{d}_k$ d is direction vector
 η is step size in the direction of minimum
 k is the current iteration number

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

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

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

The series $\{\theta_k\}_{k=1,2,\dots}$ should converge to a local minimum θ

- 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 $E(\theta_{\text{next}}) = E(\theta_{\text{now}} + \eta d) < E(\theta_{\text{now}})$
- The principal differences between various descent algorithms lie in the first procedure for determining successive directions

Once d is determined, η^* is computed as:

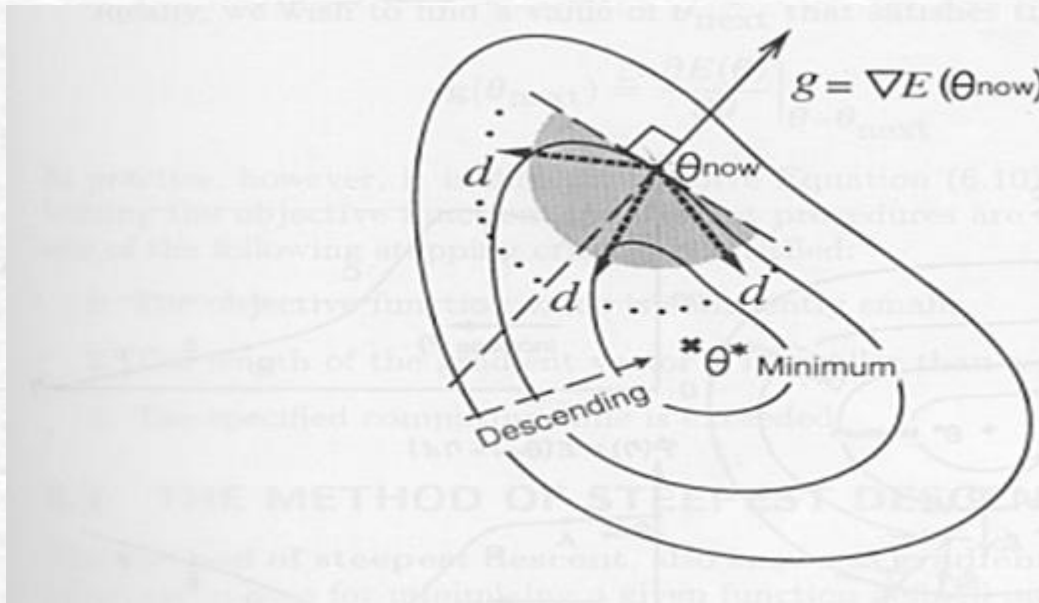
$$\eta^* = \arg \min_{\eta > 0} \mathcal{J}(\eta)$$

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

3.2.2. Gradient-based methods

● **Definition:** The gradient of a differentiable function $E: \mathbb{R}^n \rightarrow \mathbb{R}$ at θ is the vector of first derivatives of E , denoted as g . That is:

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



when $d = -g$, d is the steepest descent direction at a local point θ_{now}

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

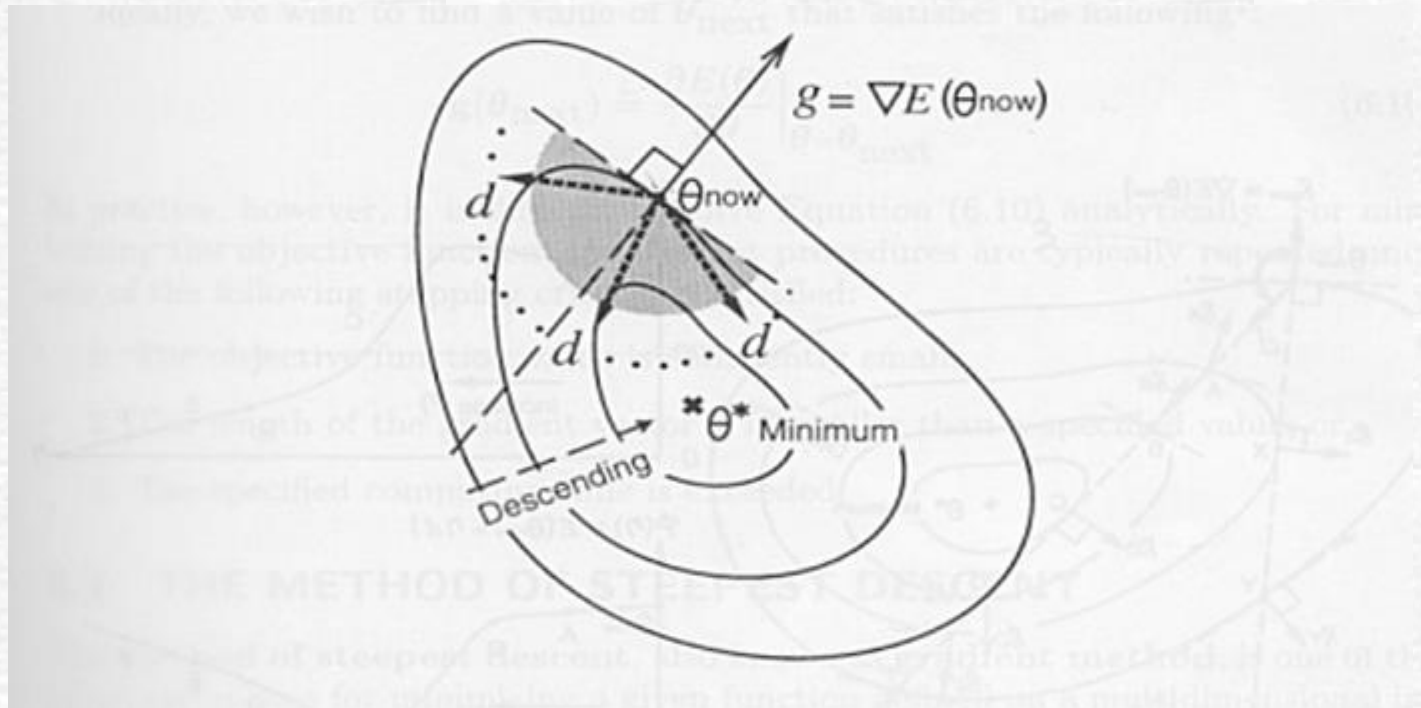
$$\phi'(\eta) = \frac{dE(\theta_{now} + \eta d)}{d\eta} \Big|_{\eta=0} = g^T d = \|g^T\| \|d\| \cos(\xi(\theta_{now})) < 0$$

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}

The previous equation is justified by Taylor series expansion:

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

For a small positive η , the inequality holds when $g^T d < 0$



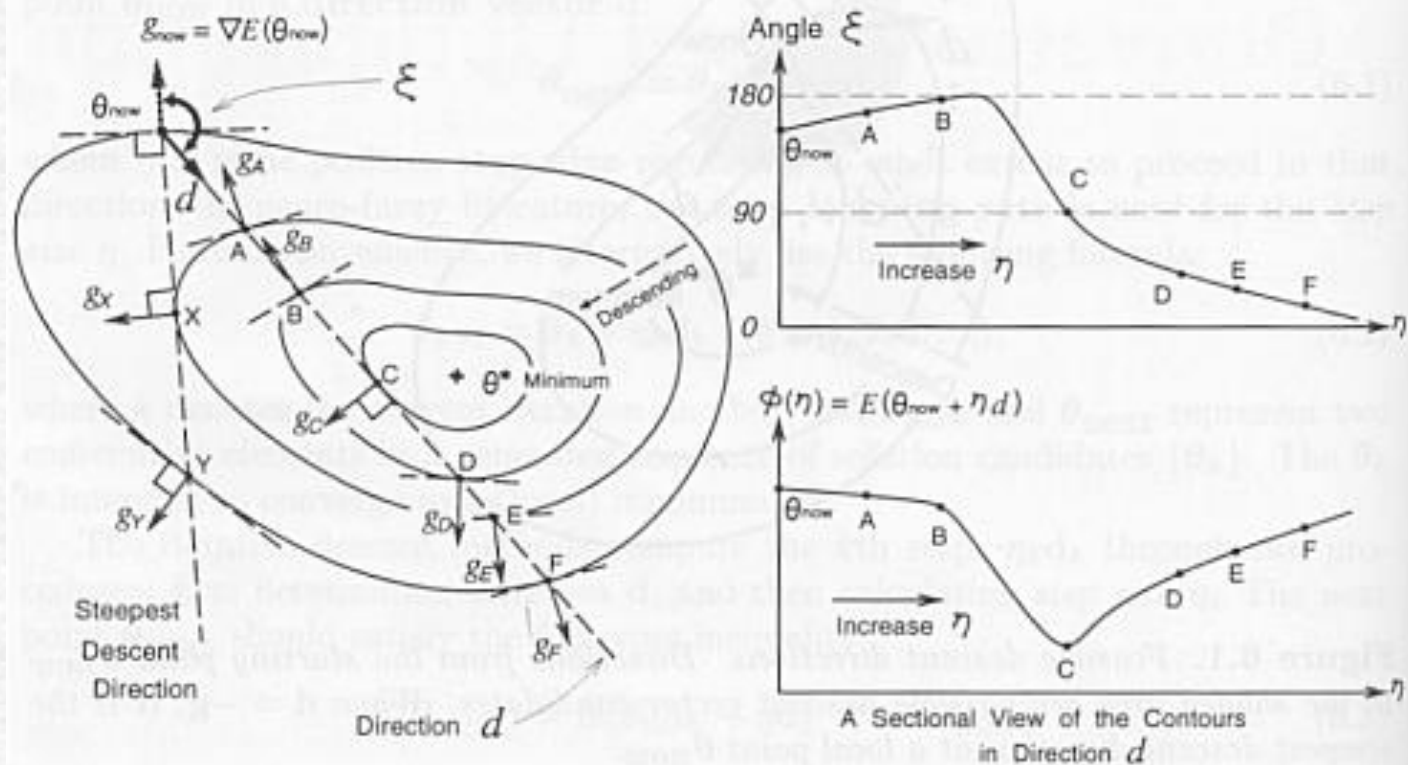


Figure 6.2. Angle ξ between gradient directions g and a descent direction d , which is determined by a certain algorithm at the current point θ_{now} . Let N be the set of all possible next points; $N \supset \{A, B, C, D, E, F, X, Y\}$. In the one-way downhill direction d , the next point θ_{next} may 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.

- 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:
$$\theta_{\text{next}} = \theta_{\text{now}} - \eta G g \quad (\eta > 0, G \text{ positive def. mat.})$$

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

$$\mathbf{g}(\theta_{\text{next}}) = \left. \frac{\partial \mathbf{E}(\theta)}{\partial \theta} \right|_{\theta=\theta_{\text{next}}} = \mathbf{0}$$

but this is difficult to solve (Necessary condition but not sufficient!)

- But practically, we stop the algorithm if:

- The objective function value is sufficiently small
- or ● The length of the gradient vector \mathbf{g} is smaller than a threshold
- or ● The computation time is exceeded

1. The method of Steepest Descent

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:

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

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}

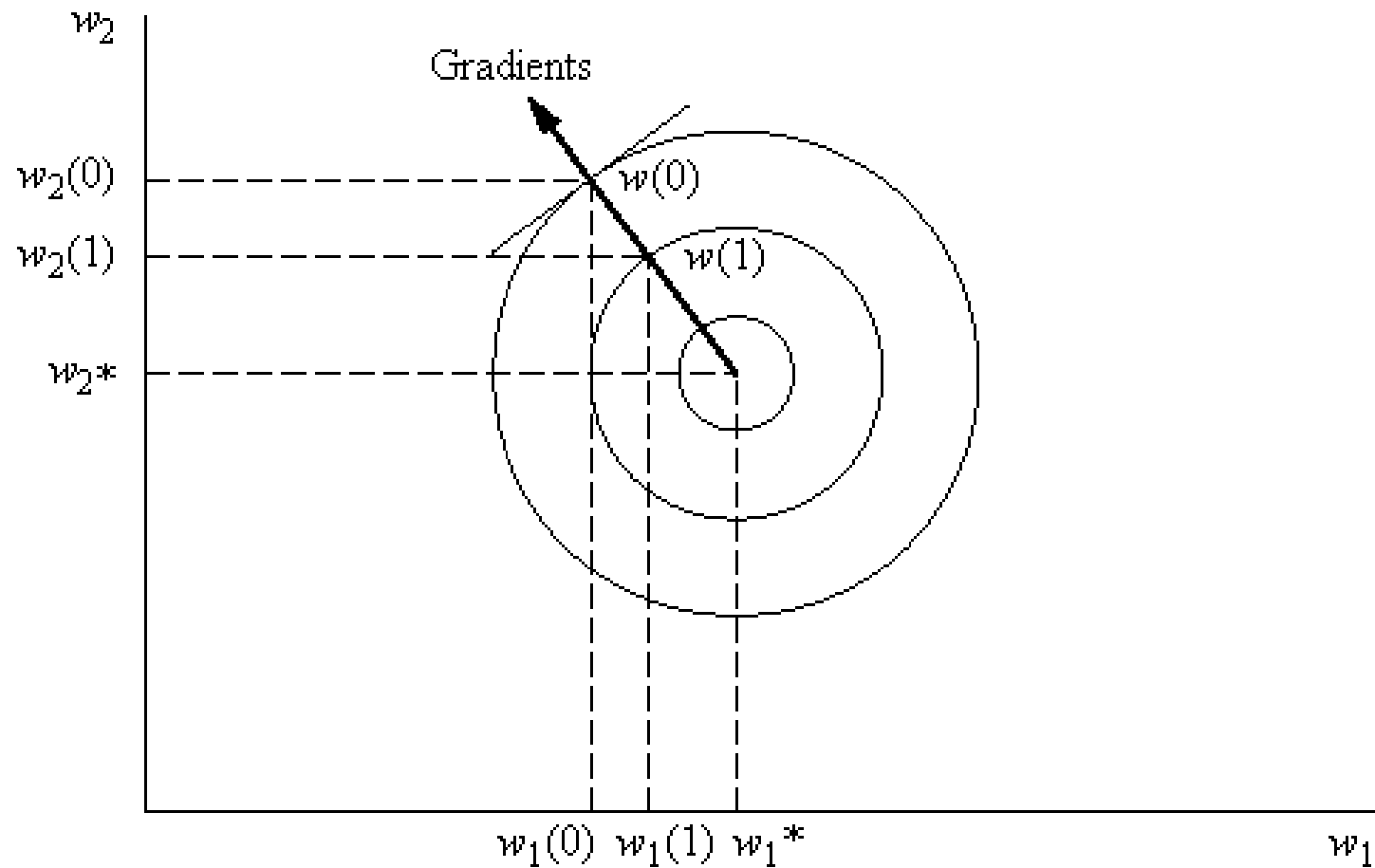
- Therefore, the negative gradient direction ($-g$) points to the locally steepest downhill direction
- This direction may not be a shortcut to reach the minimum point θ^*
- However, if the steepest descent uses the line minimization technique ($\min \phi(\eta)$) then $\phi'(\eta) = 0$
$$\phi'(\eta) = \frac{dE(\theta_{\text{now}} - \eta g_{\text{now}})}{d\eta} = \nabla^T E(\theta_{\text{now}} - \eta g_{\text{now}}) g_{\text{now}}$$
$$= g_{\text{next}}^T - g_{\text{now}}^T = 0$$

(Necessary Condition for ϕ)

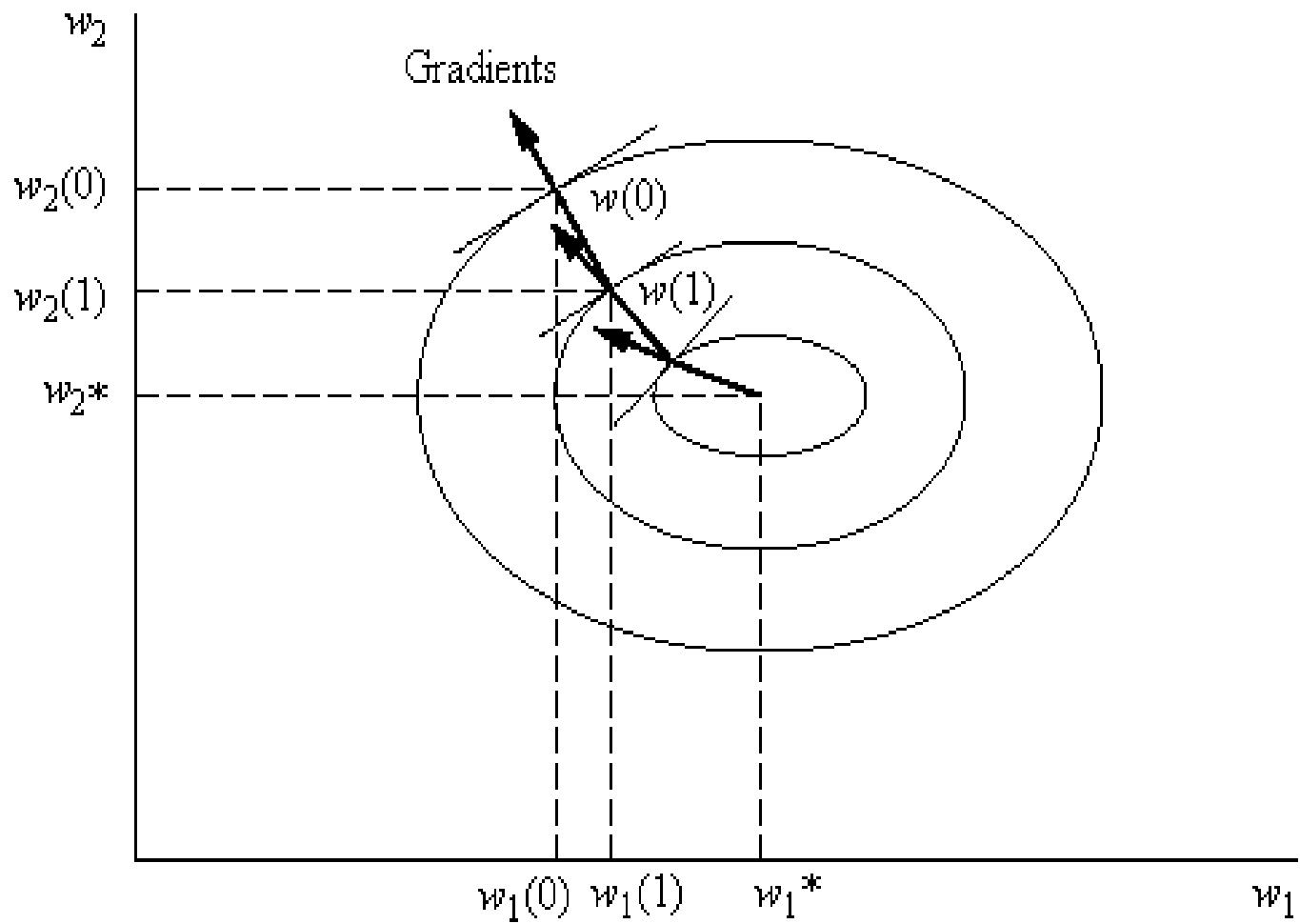
$$\Rightarrow g_{\text{next}} \text{ is orthogonal to the current gradient vector } g_{\text{now}}$$

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

The method of Steepest Descent



(a)




(b)

Weight track toward the minimum: (a) equal eigenvalues; (b) unequal eigenvalues



The descent procedures are repeated until one of the following stopping criteria is satisfied

- *The objective function value is sufficiently small,
 - *The length of the gradient vector g is smaller than a specified value,
 - *The specified computing time is exceeded.
- 

2. Newton's Methods (NM)

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:

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

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

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

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

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

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

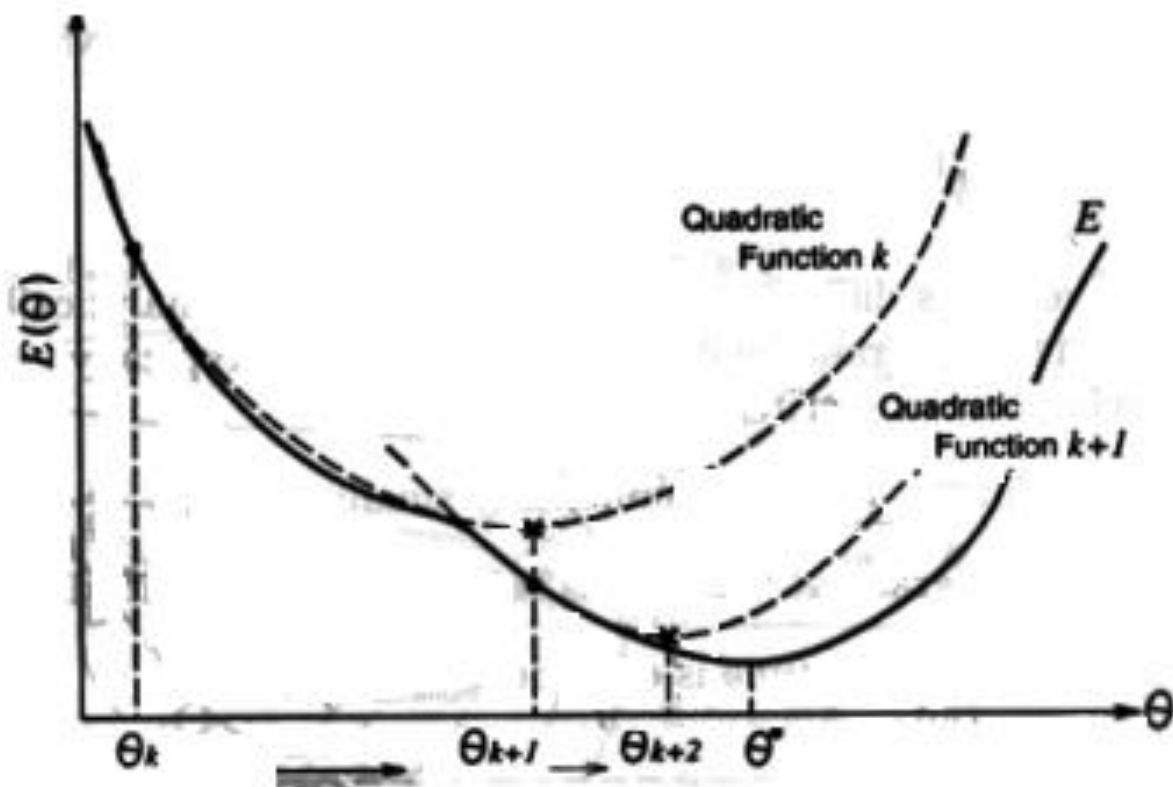


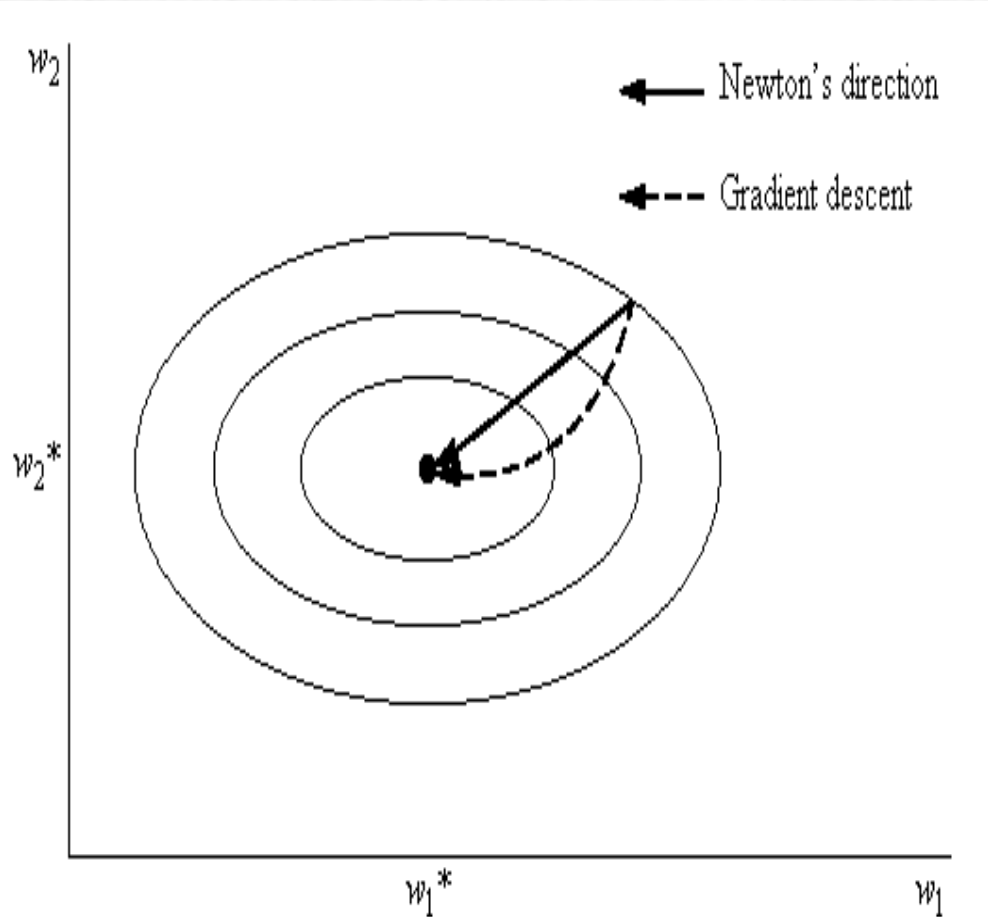
Figure 6.3. *Newton's (or Newton-Raphson) method for minimizing a general objective function E , which is approximated locally as a quadratic form; this approximate function is minimized exactly.*

- 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

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

- 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 not be reached in a single step & NM should be iteratively repeated

Directions of Searches



Directions of the steepest descent and Newton's method

- Newton's method corrects the direction of the search such that it always points to the minimum, while the gradient descent points to the maximum direction of change. These two directions may or may not coincide

Modified Newton's Method

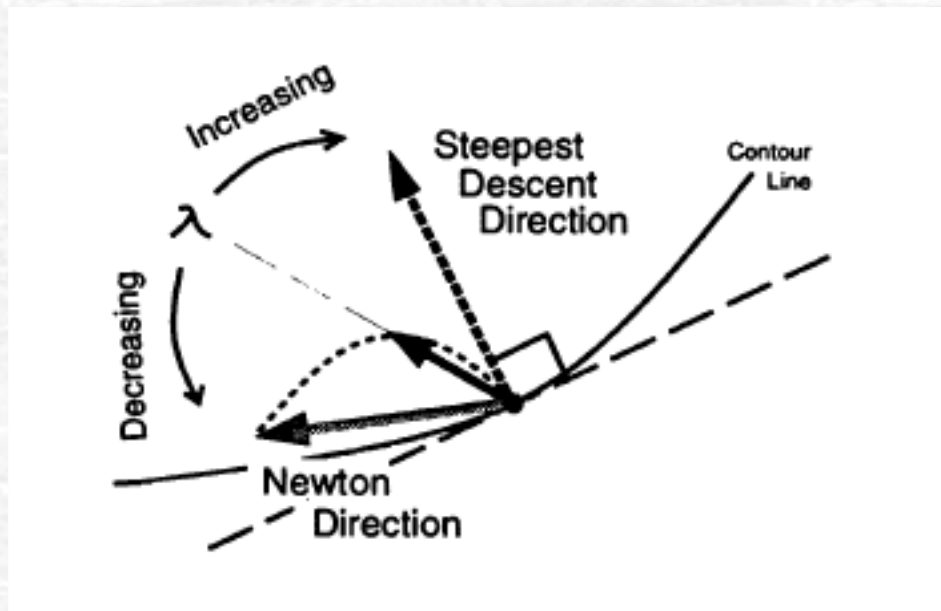
1. Adaptive Step Length

2. Levenberg- Marquardt Modifications

- ✿ The problems with Newton's method are (a) inability to distinguish maxima from minima, and
- ✿ (b) poor global convergence properties (a saddle point?).
- ✿ Both problems can be solved effectively through a restricted step suboptimization. A condition for a minimum of $E(\theta_{\text{next}})$ is that \mathbf{H} be positive definite. We therefore add a diagonal matrix to \mathbf{H} to ensure that it is positive definite

$$\theta_{k+1} = \theta_k + [\mathbf{H} + \lambda \mathbf{I}]^{-1} \mathbf{g}$$

$$\theta_{k+1} = \theta_k + \eta [\mathbf{H} + \lambda \mathbf{I}]^{-1} \mathbf{g}$$



3. Quasi-Newton method

- One of the drawbacks of Newton's method is that it requires the inverse of **H matrix** at each iteration. This is a problem if the inverse is very expensive or difficult to compute. In such cases it may be convenient to iterate according to
- $\theta_{k+1} = \theta_k + M_k^{-1} (g_{k+1} - g_k)$
- where **M_k** is an easily computed approximation to
- H_k**. For example, in one dimension, the **secant**
- method** approximates the derivative with the difference quotient
- $M_{k+1} = g_{k+1} - g_k / \theta_{k+1} - \theta_k$

$$\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k = \mathbf{M}_{k+1}(\mathbf{g}_{k+1} - \mathbf{g}_k)$$

$$\Delta\boldsymbol{\theta}_k = \mathbf{M}_{k+1}\Delta\mathbf{g}_k.$$

DFP formula

$$\mathbf{M}_{k+1} = \mathbf{M}_k + \frac{\Delta\boldsymbol{\theta}_k\Delta\boldsymbol{\theta}_k^T}{\Delta\boldsymbol{\theta}_k^T\Delta\mathbf{g}_k} - \frac{\mathbf{M}_k\Delta\mathbf{g}_k\Delta\mathbf{g}_k^T\mathbf{M}_k}{\Delta\mathbf{g}_k^T\mathbf{M}_k\Delta\mathbf{g}_k},$$

BFGS formula

$$\begin{aligned}\mathbf{M}_{k+1} &= \mathbf{M}_k + \left(1 + \frac{\Delta\mathbf{g}_k^T\mathbf{M}_k\Delta\mathbf{g}_k}{\Delta\boldsymbol{\theta}_k^T\Delta\mathbf{g}_k}\right) \frac{\Delta\boldsymbol{\theta}_k\Delta\boldsymbol{\theta}_k^T}{\Delta\boldsymbol{\theta}_k^T\Delta\mathbf{g}_k} - \frac{\Delta\boldsymbol{\theta}_k\Delta\mathbf{g}_k^T\mathbf{M}_k + \mathbf{M}_k\Delta\mathbf{g}_k\Delta\boldsymbol{\theta}_k^T}{\Delta\boldsymbol{\theta}_k^T\Delta\mathbf{g}_k} \\ &= \left(\mathbf{I} - \frac{\Delta\boldsymbol{\theta}_k\Delta\mathbf{g}_k^T}{\Delta\boldsymbol{\theta}_k^T\Delta\mathbf{g}_k}\right)\mathbf{M}_k\left(\mathbf{I} - \frac{\Delta\mathbf{g}_k\Delta\boldsymbol{\theta}_k^T}{\Delta\boldsymbol{\theta}_k^T\Delta\mathbf{g}_k}\right) + \frac{\Delta\boldsymbol{\theta}_k\Delta\boldsymbol{\theta}_k^T}{\Delta\boldsymbol{\theta}_k^T\Delta\mathbf{g}_k}\end{aligned}$$

Symmetric Rank 1 SR1

$$y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$$

$$H_{k+1} = H_k + \frac{(\Delta x_k - H_k y_k)(\Delta x_k - H_k y_k)^T}{(\Delta x_k - H_k y_k)^T y_k}$$

4. Step Size Determination

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

$$\theta_{\text{next}} = \theta_{\text{now}} + \eta d = \theta_{\text{now}} - \eta Gg$$

- This formula entails effectively determining the step size η
- $\phi'(\eta) = 0$ with $\phi(\eta) = E(\theta_{\text{now}} + \eta d)$ is often impossible to solve

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
 1. A scheme, by function evaluation for finding three points to satisfy:
$$E'(\theta_{k-1}) > E(\theta_k) < E(\theta_{k+1}); \theta_{k-1} < \theta_k < \theta_{k+1}$$
 2. 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 $h \in \mathbb{R}$, let θ_1 be $\theta_0 + h$.

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.

- SD-Step Size (Optimal)

$$E(\boldsymbol{\theta}) = \frac{1}{2} \boldsymbol{\theta}^T \mathbf{A} \boldsymbol{\theta} + \mathbf{b}^T \boldsymbol{\theta} + c.$$

$$\mathbf{g}(\boldsymbol{\theta}) = \mathbf{A} \boldsymbol{\theta} + \mathbf{b}.$$

$$\phi'(\eta^*) = \frac{dE(\boldsymbol{\theta}_{\text{now}} + \eta \mathbf{d})}{d\eta} = [\mathbf{A}(\boldsymbol{\theta} + \eta^* \mathbf{d}) + \mathbf{b}]^T (\mathbf{d}) = 0.$$

$$\eta^* = -\frac{\mathbf{d}^T \mathbf{g}}{\mathbf{d}^T \mathbf{A} \mathbf{d}}.$$

$$\boldsymbol{\theta}_{\text{next}} = \boldsymbol{\theta}_{\text{now}} - \frac{\mathbf{g}^T \mathbf{G} \mathbf{g}}{\mathbf{g}^T \mathbf{G} \mathbf{A} \mathbf{G} \mathbf{g}} \mathbf{g}.$$

Line searches

- The process of determining η^* that minimizes a one-dimensional function $\phi(\eta)$ 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 $\phi(\eta_k)$, $\phi'(\eta_k)$, and $\phi''(\eta_k)$ are available, the classical Newton method (defined by $\hat{\theta} = \theta_{now} - H^{-1}g$) can be applied to solving the equation $\phi'(\eta_k) = 0$:

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

- Secant method

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

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

this method is called the secant method.

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

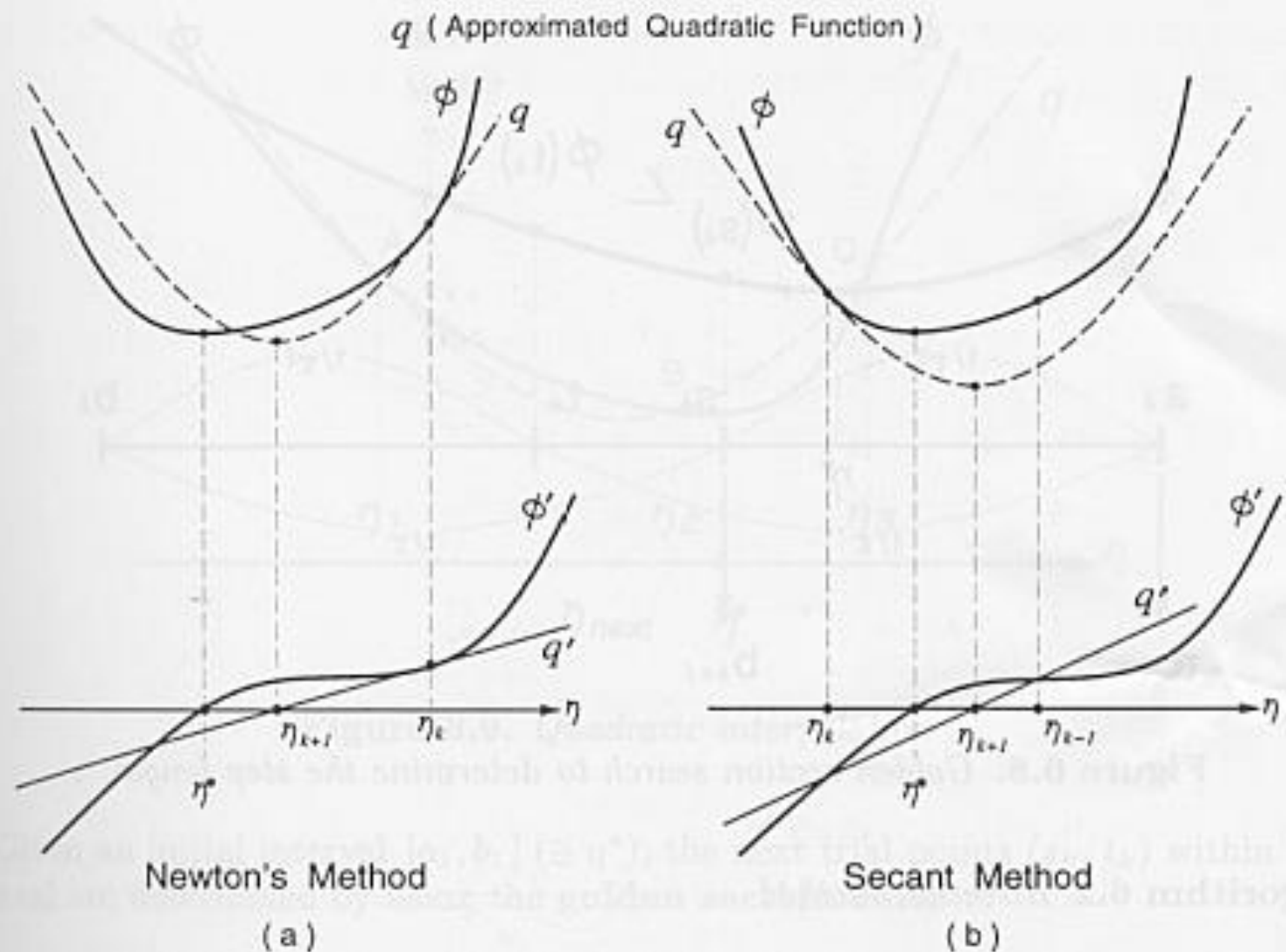


Figure 6.7. Newton's method (left) and the secant method (right) to determine the step size.

- Sectioning methods

- It starts with an interval $[a_1, b_1]$ in which the minimum η^* must lie, and then reduces the length of the interval at each iteration by evaluating the value of \emptyset at a certain number of points
- The two endpoints a_1 and b_1 can be found by the initial bracketing described previously
- The bisection method is one of the simplest sectioning method for solving $\emptyset'(\eta^*) = 0$, if first derivatives are available!

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

Algorithm [bisection method]

(1) Given $\varepsilon \in \mathbb{R}^+$ and an initial interval with 2 endpoints a_1 and

a_2 such that: $a_1 < a_2$ and $\phi(a_1)\phi(a_2) < 0$ then set:

$$\eta_{\text{left}} \leftarrow a_1$$

$$\eta_{\text{right}} \leftarrow a_2$$

(2) Compute the midpoint η_{mid} ; $\eta_{\text{mid}} \leftarrow (\eta_{\text{right}} + \eta_{\text{left}}) / 2$

if $\phi(\eta_{\text{right}}) \phi(\eta_{\text{mid}}) < 0$, $\eta_{\text{left}} \leftarrow \eta_{\text{mid}}$

Otherwise

$$\eta_{\text{right}} \leftarrow \eta_{\text{mid}}$$

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

Golden search method

This method does not require ϕ to be differentiable. Given an initial interval $[a_1, b_1]$ that contains η^* , the next trial points (s_k, t_k) within the interval are determined by using the golden section ratio τ :

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

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

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

This procedure guarantees the following:

$$a_k < s_k < t_k < b_k$$

The algorithm generates a sequence of two endpoints a_k and b_k , according to:

$$\text{If } \varnothing(s_k) > \varnothing(t_k), a_{k+1} = s_k, b_{k+1} = b_k$$

$$\text{Otherwise } a_{k+1} = a_k, b_{k+1} = t_k$$

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

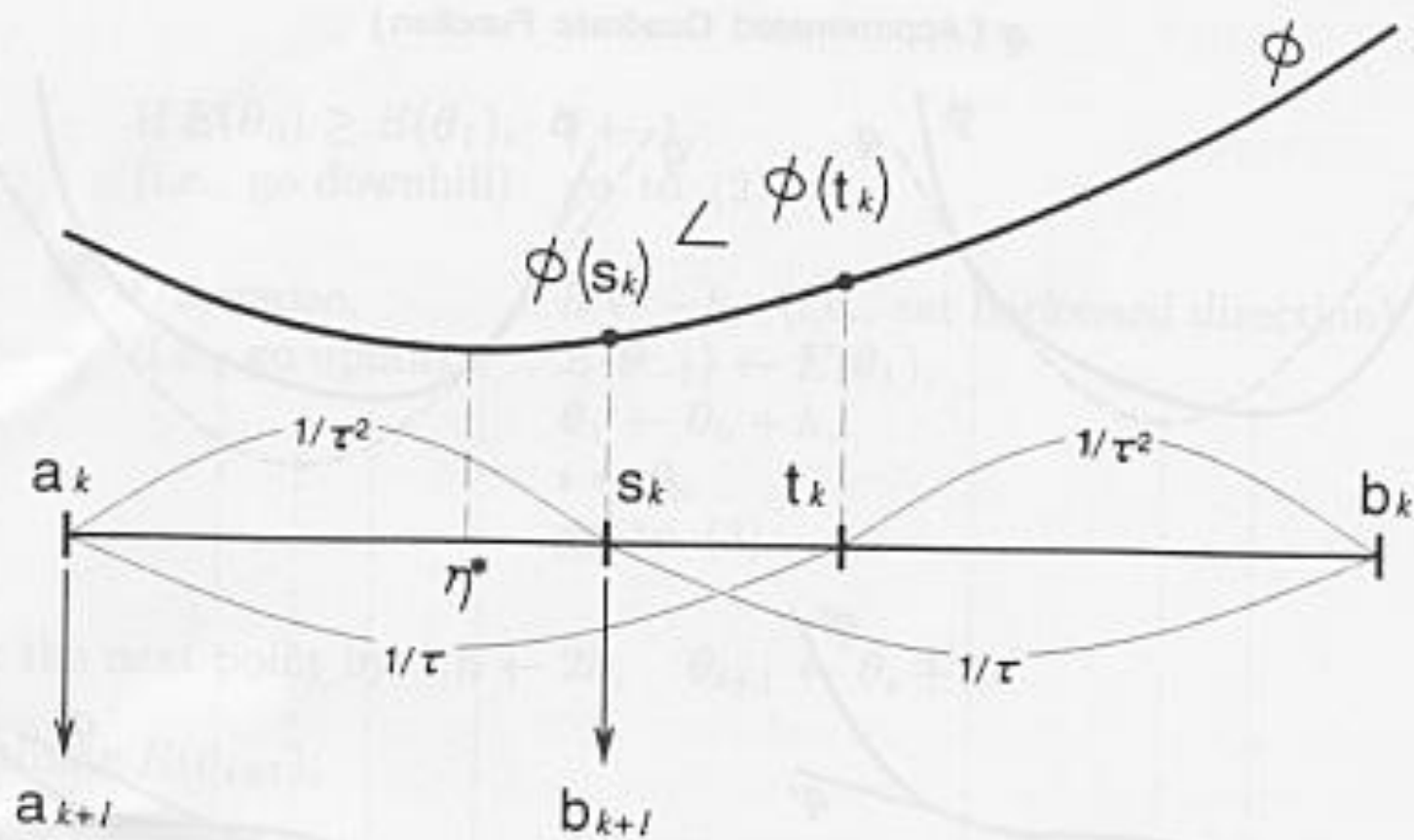


Figure 6.8. Golden section search to determine the step length. .

● 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 (η_1, \varnothing_1) , (η_2, \varnothing_2) and (η_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 $\varnothing_i = \varnothing(\eta_i)$, $i = 1, 2, 3$

- Condition for obtaining a unique minimum point is:

$q'(\eta) = 0$, therefore the next point η_{next} is:

$$\eta_{\text{next}} = \frac{1}{2} * \frac{(\eta_2^2 - \eta_3^2)\emptyset_1 + (\eta_3^2 - \eta_1^2)\emptyset_2 + (\eta_1^2 - \eta_2^2)\emptyset_3}{(\eta_2 - \eta_3)\emptyset_1 + (\eta_3 - \eta_1)\emptyset_2 + (\eta_1 - \eta_2)\emptyset_3}$$

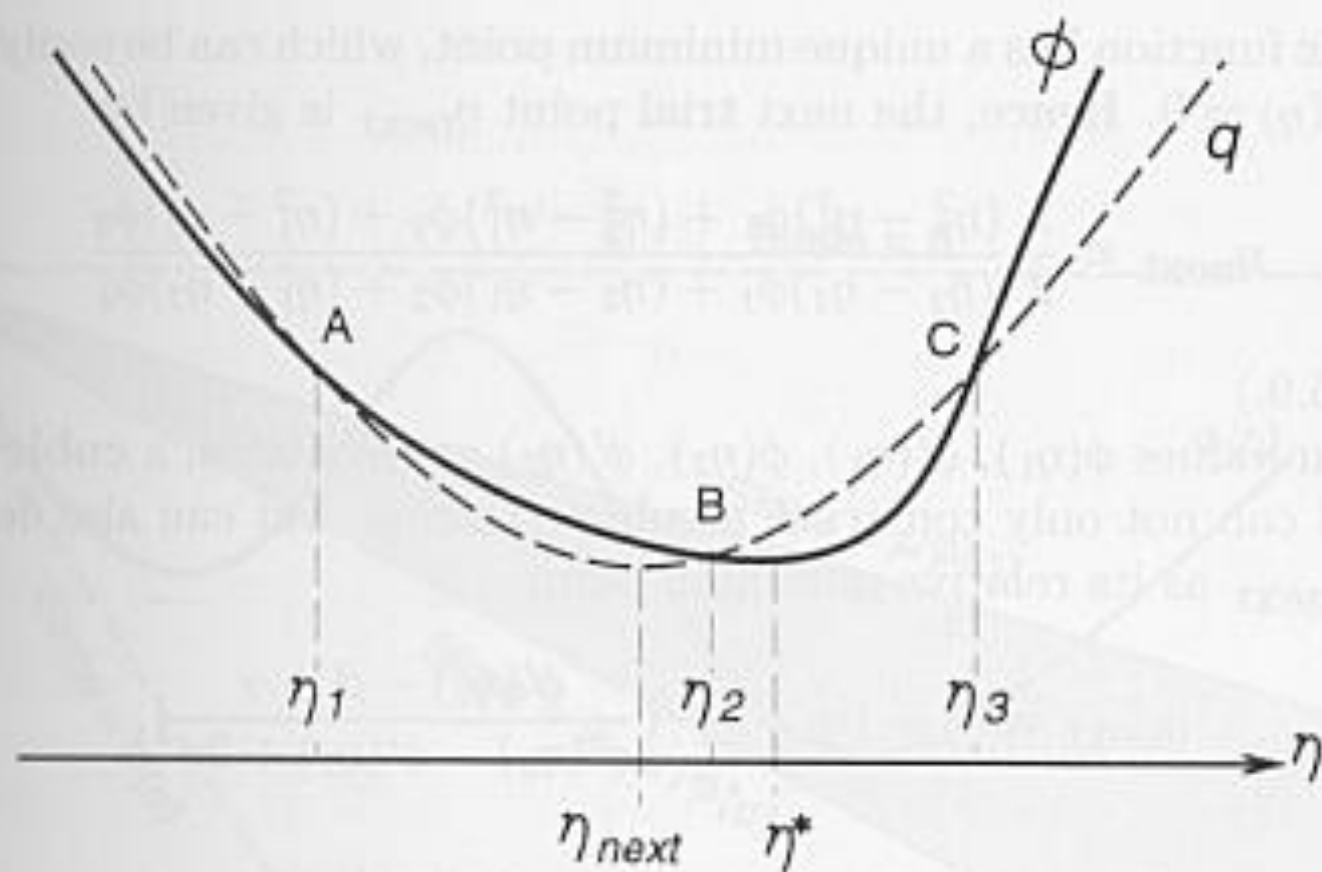


Figure 6.9. Quadratic interpolation .

Conjugate Gradient Algorithm

Generally, given a symmetric ($n \times n$) matrix \mathbf{Q} , two n -dimensional (direction) vectors \mathbf{d}_j and \mathbf{d}_k are mutually **conjugate with respect to \mathbf{Q}** , or **\mathbf{Q} -orthogonal**, if the following equation holds:

$$\mathbf{d}_j^T \mathbf{Q} \mathbf{d}_k = 0.$$

$$\mathbf{d}_k = -\mathbf{g}_k + \sum_{j=0}^{k-1} \frac{\mathbf{d}_j^T \mathbf{H} \mathbf{g}_k}{\mathbf{d}_j^T \mathbf{H} \mathbf{d}_j} \mathbf{d}_j.$$

$$\alpha_{kj} = \frac{\mathbf{d}_j^T \mathbf{H} \mathbf{g}_k}{\mathbf{d}_j^T \mathbf{H} \mathbf{d}_j}.$$

$$\alpha_{kj} = \sum_{j=0}^{k-1} \frac{\mathbf{g}_k^T (\mathbf{g}_{j+1} - \mathbf{g}_j)}{\mathbf{d}_j^T (\mathbf{g}_{j+1} - \mathbf{g}_j)} \mathbf{d}_j.$$

$$\mathbf{d}_k = -\mathbf{g}_k + \beta_k \mathbf{d}_{k-1},$$

$$\beta_k = \frac{\mathbf{g}_k^T (\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{d}_{k-1}^T (\mathbf{g}_k - \mathbf{g}_{k-1})}.$$

$$\alpha_{kj} = \begin{cases} 0 & \text{if } j < k-1 \\ \alpha_{k,k-1} & \text{otherwise,} \end{cases}$$

$$\alpha_{k,k-1} = \frac{\mathbf{g}_k^T (\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{d}_{k-1}^T (\mathbf{g}_k - \mathbf{g}_{k-1})} \mathbf{d}_k.$$

Choosing the term β

Fletcher-Reeves: $\beta_k = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}}$

Polak-Ribiere: $\beta_k = \frac{\mathbf{g}_k^T (\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}}$

Beale-Sorenson: $\beta_k = \frac{\mathbf{g}_k^T (\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{d}_{k-1}^T (\mathbf{g}_k^T - \mathbf{g}_{k-1}^T)}$

Nonlinear Least-Squares Problems

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 $(x_p; t_p)$, $(p = 1, \dots, m)$, we can write:

$$\begin{aligned} E(\theta) &= \sum_{p=1}^m (t_p - y_p)^2 = \sum_{p=1}^m (t_p - f(x_p, \theta))^2 \\ &= \sum_{p=1}^m r_p(\theta)^2 = \mathbf{r}^T(\theta) \cdot \mathbf{r}(\theta) \end{aligned}$$

The gradient is expressed as:

$$\mathbf{g} = \mathbf{g}(\boldsymbol{\theta}) = \frac{\partial \mathbf{E}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 2 \sum_{p=1}^m \mathbf{r}_p(\boldsymbol{\theta}) \frac{\partial \mathbf{r}_p(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 2 \mathbf{J}^T \cdot \mathbf{r}$$

where \mathbf{J} is the Jacobian matrix of \mathbf{r} .

Since $\mathbf{r}_p(\boldsymbol{\theta}) = \mathbf{t}_p - \mathbf{f}(\mathbf{x}_p, \boldsymbol{\theta})$, this implies that the p th row of \mathbf{J} is: $-\nabla_{\boldsymbol{\theta}}^T \mathbf{f}(\mathbf{x}_p, \boldsymbol{\theta})$

Hessian

$$\begin{aligned} \mathbf{H} = \mathbf{H}(\boldsymbol{\theta}) &\equiv \frac{\partial^2 \mathbf{E}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T} \\ &= 2 \sum_{p=1}^m \left[\frac{\partial \mathbf{r}_p(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \frac{\partial \mathbf{r}_p(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^T} + \mathbf{r}_p(\boldsymbol{\theta}) \frac{\partial^2 \mathbf{r}_p(\boldsymbol{\theta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T} \right] \\ &= 2(\mathbf{J}^T \mathbf{J} + \mathbf{S}), \end{aligned}$$

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 to obtain the model parameters
- The parameters $\theta^T = (\theta_1, \theta_2, \dots, \theta_n, \dots)$ will be computed iteratively
- Taylor expansion of $y = f(x, \theta)$ around $\theta = \theta_{\text{now}}$

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

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

$$\begin{aligned} E(\theta) &= \left\| t - f(x, \theta_{\text{now}}) - \frac{\partial f(x, \theta_{\text{now}})}{\partial \theta} (\theta - \theta_{\text{now}}) \right\|^2 \\ &= \left\| r + J^T (\theta - \theta_{\text{now}}) \right\|^2 = \left\| r + J^T d \right\|^2 \end{aligned}$$

where $d = \theta - \theta_{\text{now}}$

➤ The next point θ_{next} is obtained by:

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

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

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

(since $\mathbf{g} = 2\mathbf{J}^T \mathbf{r}$)

Least-Square Methods for System Identification

- Statistical Properties & the Maximum Likelihood Estimator
- LSE for Nonlinear Models

Statistical Properties & the Maximum Likelihood Estimator

Statistical qualities of LSE

● *Definition [unbiased estimator]*

An estimator θ^+ of the parameter θ is unbiased if $E(\theta^+) = \theta$, where $E[.]$ is the statistical expectation

● *Definition [minimal variance]*

An estimator θ^+ is a minimum variance estimator if for any other estimator θ^* : $\text{cov}(\theta^+) \leq \text{cov}(\theta^*)$, where $\text{cov}()$ is the covariance matrix of the random vector θ .

Maximum likelihood (ML) estimator

- ML is one of the most widely used technique for parameter estimation of a statistical distribution

- *ML definition:*

For a sample of n observations (of a probability density function) x_1, x_2, \dots, x_n , the likelihood function L is defined by:

$$L = f(x_1; \theta) f(x_2; \theta) \dots f(x_n; \theta)$$

- The criterion for choosing θ is:

“pick a value of θ that provides a high probability of obtaining the actual observed data x_1, x_2, \dots, x_n ”

- Therefore, ML estimator θ^+ is defined as the value of θ which maximizes L:
$$\frac{\partial \mathbf{L}}{\partial \theta}(\hat{\theta}) = \mathbf{0}$$

or equivalently:

$$\frac{\partial \ln \mathbf{L}}{\partial \theta}(\hat{\theta}) = \mathbf{0}$$

Example 1: ML estimation for exponential distribution

$$f(x; \theta) = \theta^{-1} e^{-x/\theta}$$

$$\begin{aligned} L &= (\theta^{-1} e^{-x_1/\theta}) (\theta^{-1} e^{-x_2/\theta}) \dots (\theta^{-1} e^{-x_m/\theta}) \\ &= \theta^{-m} \exp\left(-\theta^{-1} \sum x_i\right) \end{aligned}$$

is the likelihood function for m observations x_1, x_2, \dots, x_m .

$$\ln L = -m \ln \theta - \frac{1}{\theta} \sum_{i=1}^m x_i \qquad \frac{\partial \ln L}{\partial \theta} = -\frac{m}{\theta} + \frac{\sum x_i}{\theta^2} = 0$$

$$\text{Therefore : } \hat{\theta} = \frac{\sum x_i}{m}$$

Example 2: ML estimation for normal distribution

$$f(\mathbf{x}; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{\mathbf{x} - \mu}{\sigma}\right)^2\right]$$

where μ and σ^2 are respectively the mean & the variance unknown parameters.

For m observations x_1, x_2, \dots, x_m , we have:

$$L = \left(\frac{1}{\sqrt{2\pi}\sigma} \right)^m \exp \left[-\frac{1}{2\sigma^2} (\mathbf{x}_i - \mu)^2 \right]$$

$$\text{and } \ln L = -m * \ln(\sqrt{2\pi}\sigma) - \frac{1}{2\sigma^2} \sum (\mathbf{x}_i - \mu)^2$$

$$\text{Therefore : } \frac{\partial \ln L}{\partial \mu} = 0 \Rightarrow \frac{1}{\sigma^2} \sum (\mathbf{x}_i - \mu) = 0 \Rightarrow \hat{\mu} = \frac{\sum \mathbf{x}_i}{m}$$

$$\frac{\partial \ln L}{\partial \sigma} = 0 \Rightarrow -\frac{m}{\sigma} - \frac{1}{2} (-2\sigma^{-3}) \sum (\mathbf{x}_i - \mu)^2 = 0$$

$$\Rightarrow \hat{\sigma}^2 = \frac{\sum (\mathbf{x}_i - \hat{\mu})^2}{m}$$

LSE for Nonlinear Models

- Nonlinear models are divided into 2 families
 - Intrinsically linear
 - Intrinsically nonlinear
 - Through appropriate transformations of the input-output variables & fitting parameters, an intrinsically linear model can become a linear model
 - By this transformation into linear models, LSE can be used to optimize the unknown parameters

Example: Decay of the radioactivity of chemicals

$$y = ae^{bt} \quad (a, b < 0 \text{ are to be determined})$$

(t, y) are the input-output data

given a training data $\{(t_i, y_i), i = 1, \dots, m\}$

$$E(a, b) = \sum_{i=1}^m (y_i - ae^{bt_i})^2$$

minimizing the error yields
$$\begin{cases} \frac{\partial E}{\partial a} = 2 \sum_{i=1}^m (y_i - ae^{-bt_i}) (-e^{bt_i}) = 0 \\ \frac{\partial E}{\partial b} = 2 \sum_{i=1}^m (y_i - ae^{-bt_i}) (-at_i e^{-bt_i}) = 0 \end{cases}$$

we obtain 2 nonlinear equations with respect to a & b ! Therefore, we are better off to linearize the model before.

$$\ln y = \ln a + bt$$

with $\ln a = a'$, $\ln y = y'$ and $b = b'$

$$\Rightarrow y' = a' + b't$$

which is an intrinsically linear model. The linear & nonlinear models are close but can sometimes differ significantly!

The following table shows a family of intrinsically linear models

Nonlinear models	Transformation	Linear forms
$y = ae^{bx}$	Natural logarithm	$\ln y = \ln a + bx$
$y = ax^b$	Natural logarithm	$\ln y = \ln a + b \ln x$
$y = \frac{ax}{b+x}$	Reciprocal	$\frac{1}{y} = \frac{1}{a} + \frac{b}{a} \frac{1}{x}$
$y = \frac{a}{b+x}$	Reciprocal	$\frac{1}{y} = \frac{1}{a} + \frac{b}{a} x$

Suppose that the underlying model is: $y = ae^{bt}$

t_i	0	0.80	1.84	2.90	4.06	4.81	6.07	7.06	8.15	8.87	9.98
y_i	0.98	0.69	0.47	0.46	0.29	0.16	0.23	0.10	0.03	0.12	0.01

