Θεωρία Βελτιστοποίησης

Βασίλειος Μαχαιράς

Πολιτικός Μηχανικός Ph.D.

Μη γραμμικός προγραμματισμός: βελτιστοποίηση χωρίς περιορισμούς

Πανεπιστήμιο Θεσσαλίας
Σχολή Θετικών Επιστημών
Τμήμα Πληροφορικής

Διάλεξη 7-8η/2017

Τι παρουσιάστηκε έως σήμερα

- Αναλυτικές μέθοδοι επίλυσης προβλημάτων βελτιστοποίησης.
- Μέθοδοι γραμμικού προγραμματισμού για την επίλυση προβλημάτων με γραμμική αντικειμενική συνάρτηση και γραμμικούς περιορισμούς ισότητας ή ανισότητας.
- Μέθοδοι επίλυσης μονοδιάστατων προβλημάτων βελτιστοποίησης, οι οποίες είναι χρήσιμες στην εύρεση του βέλτιστου μήκους βήματος σε προβλήματα επίλυσης με επαναληπτικές μεθόδους.

Τι θα παρουσιαστεί

- Μέθοδοι επίλυσης μη γραμμικών προβλημάτων.
- Η συγκεκριμένη διάλεξη θα εστιάσει σε προβλήματα χωρίς περιορισμούς.
- Ένα πρακτικό πρόβλημα σπάνια δεν έχει περιορισμούς. Όμως η μελέτη τέτοιων προβλημάτων είναι σημαντική διότι:
- οι περιορισμοί σε κάποια προβλήματα έχουν μικρή σημασία,
- μερικές από τις πιο αποδοτικές μεθόδους επίλυσης προβλημάτων με περιορισμούς κάνουν χρήση των αντίστοιχων τεχνικών επίλυσης χωρίς περιορισμούς,
- Πολλά προβλήματα με περιορισμούς μπορούν να μετασχηματιστούν σε προβλήματα χωρίς περιορισμούς,
- παρέχει τη βάση για την κατανόηση των μεθόδων εφαρμογής σε προβλήματα με περιορισμούς.

Εισαγωγή

- Η γενική περιγραφή προβλήματος βελτιστοποίησης χωρίς περιορισμούς είναι ως εξής:

$$\text{Find } \mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} \text{ which minimizes } f(\mathbf{X})$$

Εισαγωγή

- Στη 2η διάλεξη παρουσιάστηκαν οι αναλυτικές τεχνικές επίλυσης προβλημάτων βελτιστοποίησης και παρουσιάστηκε ότι ένα σημείο **X*** για να αποτελεί τοπικό ελάχιστο της f(**X**) πρέπει να ισχύει η αναγκαία συνθήκη:

$$\frac{\partial f}{\partial x_i}(\mathbf{X} = \mathbf{X}^*) = 0, \quad i = 1, 2, \ldots, n \qquad (6.2)$$

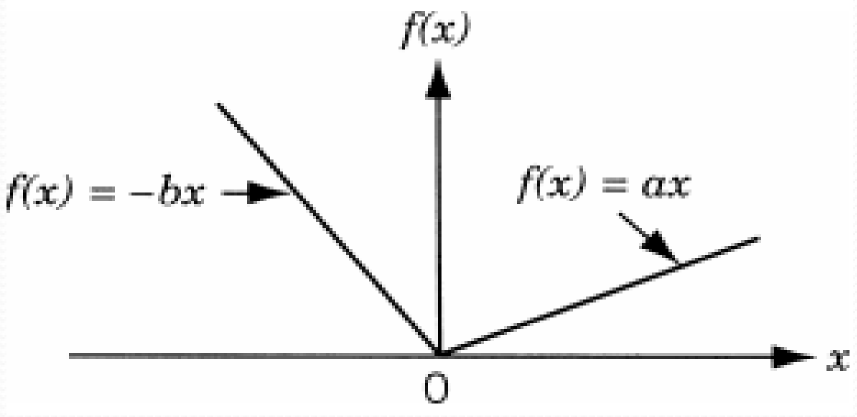
- Για να εξασφαλιστεί ότι είναι τοπικό ελάχιστο πρέπει επίσης ο Εσσιανός πίνακας (Hessian matrix) να είναι θετικά ορισμένος (positive definite):

$$\mathbf{J}_{\mathbf{X}^*} = [J]_{\mathbf{X}^*} = \left[\frac{\partial^2 f}{\partial x_i\, \partial x_j}(\mathbf{X}^*)\right] = \text{positive definite} \qquad (6.3)$$

- Οι προηγούμενες εξισώσεις 6.2-6.3 μπορούν να αναγνωρίσουν το βέλτιστο σημείο με αριθμητικές μεθόδους.
- Αν όμως η αντικειμενική συνάρτηση δεν είναι παραγωγίσιμη τότε δεν μπορούν να χρησιμοποιηθούν για να αναγνωρίσουν το βέλτιστο σημείο. Αυτό γίνεται κατανοητό με το εξής παράδειγμα: έστω ότι θέλουμε να ελαχιστοποιήσουμε το εξής:

$$f(x) = \begin{cases} ax & \text{for} \quad x \geq 0 \\ -bx & \text{for} \quad x \leq 0 \end{cases}$$

όπου α>0 και b>0



- Η συγκεκριμένη συνάρτηση δεν είναι παραγωγίσιμη στο βέλτιστο σημείο $x^*=0$, οπότε δεν μπορούν να εφαρμοστούν οι εξισώσεις 6.2-6.3 για την αναγνώρισή του.
- Σε τέτοιες περιπτώσεις για την αναγνώριση του βέλτιστου εφαρμόζεται η γενική συνθήκη: $f(\mathbf{X}^*)<f(\mathbf{X})$ για όλα τα $\mathbf{X}$.

Κατηγοριοποίηση των μεθόδων επίλυσης

- Οι υπολογιστικές τεχνικές βελτιστοποίησης προβλημάτων χωρίς περιορισμούς κατατάσσονται σε δύο κατηγορίες:
- Μέθοδοι άμεσης αναζήτησης (Direct search methods).
- Μέθοδοι καθόδου ή μέθοδοι κλίσης (Descent methods).

Μέθοδοι άμεσης αναζήτησης (Direct search methods)

- Οι μέθοδοι άμεσης αναζήτησης απαιτούν μόνο τον υπολογισμό της αντικειμενικής συνάρτησης και δεν χρησιμοποιούν τις μερικές παραγώγους της για τον εντοπισμό του ακρότατου.
- Ονομάζονται και μέθοδοι χωρίς κλίση (nongradient methods) και μέθοδοι μηδενικής τάξης (zeroth-order methods).
- Είναι κατάλληλοι για ευκολότερα προβλήματα με σχετικά λίγες μεταβλητές.
- Είναι λιγότερο αποτελεσματικές από τις μεθόδους καθόδου, αλλά εφαρμόζονται και σε συναρτήσεις στις οποίες δεν υπάρχει ή δεν είναι εύκολο να υπολογιστεί η παράγωγος.

Μέθοδοι καθόδου ή κλίσης (descent ή gradient ή indirect methods)

- Οι μέθοδοι καθόδου (ή μέθοδοι κλίσης) απαιτούν εκτός από τον υπολογισμό της αντικειμενικής συνάρτησης και τον υπολογισμό της 1ης παραγώγου ή και της 2ης, ανάλογα με τη μέθοδο.
- Αυτές οι μέθοδοι που χρησιμοποιούν μόνο την 1η παράγωγο ονομάζονται 1ης τάξης μέθοδοι. Αντίστοιχα όσες χρησιμοποιούν και τη 2η παράγωγο ονομάζονται 2ας τάξης μέθοδοι.
- Είναι αποδοτικότερες μέθοδοι σε σχέση με την άμεση αναζήτηση, λόγω των επιπλέον πληροφοριών που παρέχει ο υπολογισμός των παραγώγων.

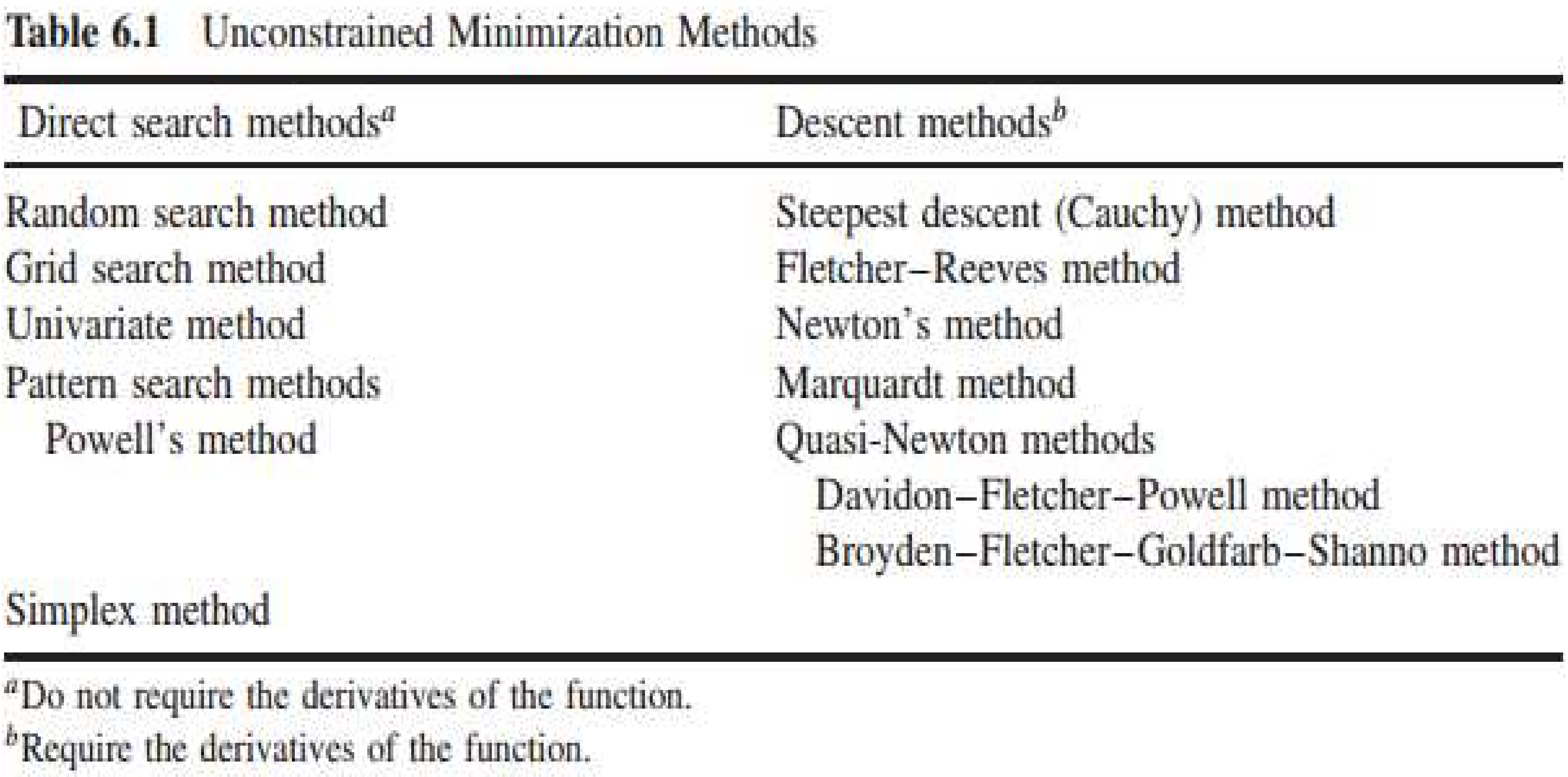


Table 6.1 Unconstrained Minimization Methods

Direct search methods[a]	Descent methods[b]
Random search method	Steepest descent (Cauchy) method
Grid search method	Fletcher–Reeves method
Univariate method	Newton's method
Pattern search methods	Marquardt method
Powell's method	Quasi-Newton methods
	Davidon–Fletcher–Powell method
	Broyden–Fletcher–Goldfarb–Shanno method
Simplex method	

[a]Do not require the derivatives of the function.

[b]Require the derivatives of the function.

Γενική προσέγγιση

- Όλες οι μέθοδοι βελτιστοποίησης προβλημάτων χωρίς περιορισμούς είναι επαναληπτικές.
- Εκκινούν από ένα δοκιμαστικό σημείο (λύση) και προχωρούν προς το ακρότατο με διαδοχικά βήματα (δοκιμές).
- Οι διαφορές ανάμεσα στις μεθόδους έγκειται:
- στο πώς επιλέγουν την κατεύθυνση αναζήτησης,
- πώς εκτιμούν το μήκος βήματος προς εκεί και
- πώς ελέγχουν το νέο σημείο για το αν είναι βέλτιστο.

Επαναληπτικές μέθοδοι

- 1. Εκκίνηση από ένα δοκιμαστικό σημείο $\mathbf{X_1}$.
- 2. Εύρεση της κατάλληλης κατεύθυνσης αναζήτησης $\mathbf{S_i}$ (ξεκινώντας με i=1), η οποία οδηγεί προς το βέλτιστο.
- 3. Εύρεση του κατάλληλου μήκους βήματος λ_i^* προς την κατεύθυνση $\mathbf{S_i}$.
- 4. Υπολογισμός του νέου σημείου $\mathbf{X}_{i+1} = \mathbf{X}_i + \lambda_i^* \mathbf{S}_i$
- 5. Έλεγχος του $\mathbf{X}_{i+1}$ αν είναι βέλτιστο. Αν είναι τότε ολοκληρώνεται η διαδικασία. Αλλιώς τίθεται $i = i + 1$ και επαναλαμβάνεται η διαδικασία στο βήμα 2.

Ρυθμός σύγκλισης (Rate of Convergence)

- Οι διάφορες μέθοδοι έχουν διαφορετικούς ρυθμούς σύγκλισης.
- Γενικώς μια μέθοδο βελτιστοποίησης αναφέρεται ότι έχει ρυθμό σύγκλισης της τάξης *p* αν:

$$\frac{\|\mathbf{X}_{i+1} - \mathbf{X}^*\|}{\|\mathbf{X}_i - \mathbf{X}^*\|^p} \le k, \quad k \ge 0, \ p \ge 1$$

- όπου **Xi** και $\mathbf{X}_{i+1}$ είναι τα σημεία που υπολογίστηκαν στην επανάληψη *i* και *i+1* αντίστοιχα, το $\mathbf{X}^*$ αντιπροσωπεύει το βέλτιστο σημείο και $\|\mathbf{X}\|$ είναι το μήκος του διανύσματος **X**:

$$\|\mathbf{X}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

- Αν p=1 και 0 ≤ k ≤ 1 τότε η μέθοδος *συγκλίνει γραμμικά* (αργά).
- Αν p=2 τότε η μέθοδος *συγκλίνει τετραγωνικά – quadratically convergent* (αντιστοιχεί σε γρήγορη σύγκλιση).
- Υπεργραμμική σύγκλιση (superlinear convergence) ονομάζεται όταν:

$$\lim_{i\to\infty} \frac{\|\mathbf{X}_{i+1} - \mathbf{X}^*\|}{\|\mathbf{X}_i - \mathbf{X}^*\|} \to 0$$

Μέθοδοι άμεσης αναζήτησης
Direct Search Methods

Μέθοδοι τυχαίας αναζήτησης (random search methods)

- Βασίζονται στη χρήση τυχαίων αριθμών για να βρουν το ελάχιστο σημείο. Η περισσότερες βιβλιοθήκες προγραμματιστικού κώδικα έχουν γεννήτριες τυχαίων αριθμών (random number generators), οπότε μπορούν να εφαρμοστούν οι τεχνικές εύκολα.
- Μερικές μέθοδοι τυχαίας αναζήτησης είναι:
- Random Jumping Method
- Random Walk Method
- Random Walk Method with Direction Exploitation

Random Jumping Method

Although the problem is an unconstrained one, we establish the bounds l_i and u_i for each design variable $x_i, i = 1, 2, \ldots, n$, for generating the random values of x_i:

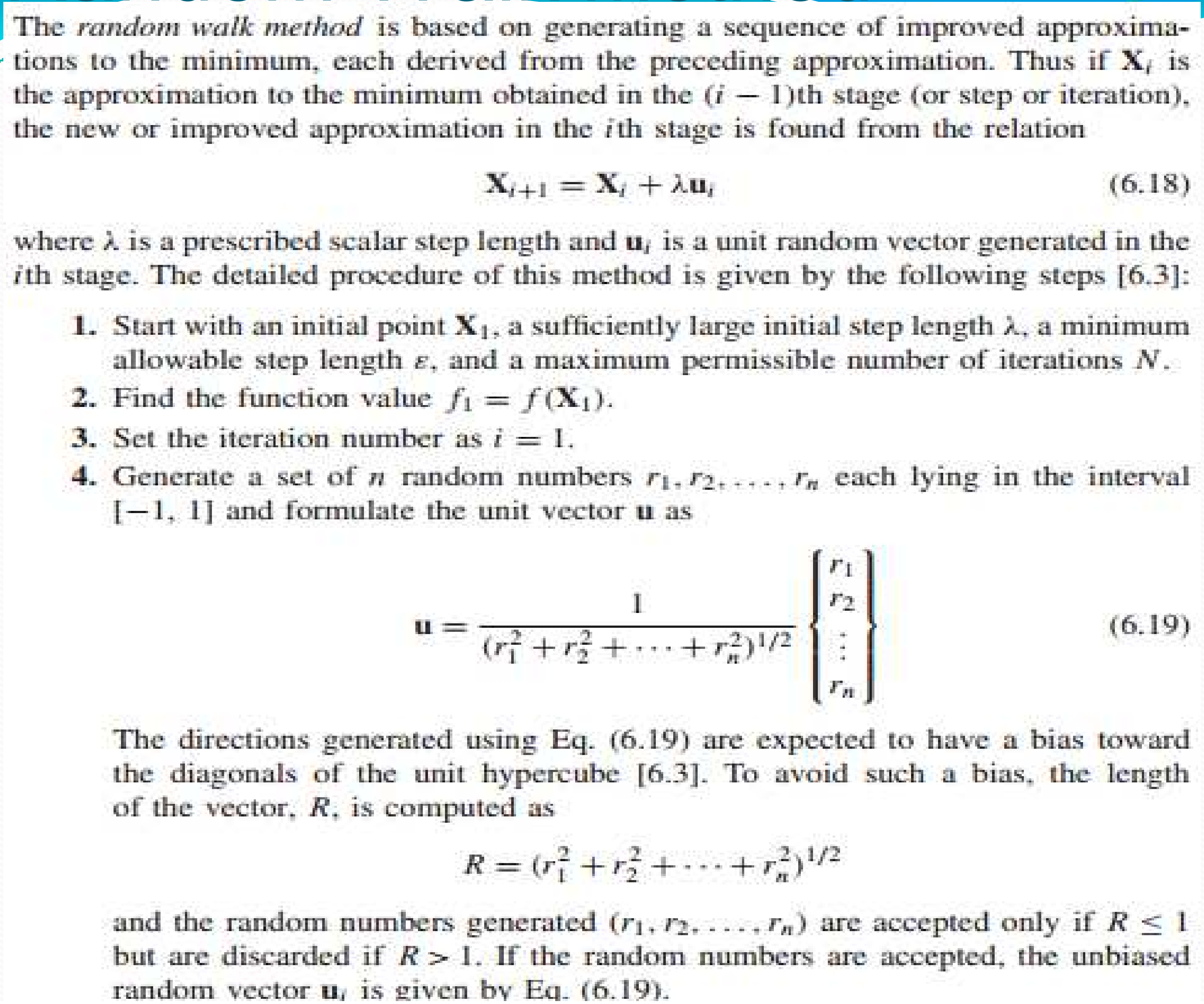
$$l_i \le x_i \le u_i, \quad i = 1, 2, \ldots, n \tag{6.16}$$

In the random jumping method, we generate sets of n random numbers, $(r_1, r_2, \ldots, r_n)$, that are uniformly distributed between 0 and 1. Each set of these numbers is used to find a point, **X**, inside the hypercube defined by Eqs. (6.16) as

$$\mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} = \begin{Bmatrix} l_1 + r_1(u_1 - l_1) \\ l_2 + r_2(u_2 - l_2) \\ \vdots \\ l_n + r_n(u_n - l_n) \end{Bmatrix} \tag{6.17}$$

and the value of the function is evaluated at this point **X**. By generating a large number of random points **X** and evaluating the value of the objective function at each of these points, we can take the smallest value of $f(\mathbf{X})$ as the desired minimum point.

Random Walk Method



The *random walk method* is based on generating a sequence of improved approximations to the minimum, each derived from the preceding approximation. Thus if $\mathbf{X}_i$ is the approximation to the minimum obtained in the $(i-1)$th stage (or step or iteration), the new or improved approximation in the ith stage is found from the relation

$$\mathbf{X}_{i+1} = \mathbf{X}_i + \lambda \mathbf{u}_i \tag{6.18}$$

where λ is a prescribed scalar step length and $\mathbf{u}_i$ is a unit random vector generated in the ith stage. The detailed procedure of this method is given by the following steps [6.3]:

1. Start with an initial point $\mathbf{X}_1$, a sufficiently large initial step length λ, a minimum allowable step length ε, and a maximum permissible number of iterations N.
2. Find the function value $f_1 = f(\mathbf{X}_1)$.
3. Set the iteration number as $i = 1$.
4. Generate a set of n random numbers $r_1, r_2, \ldots, r_n$ each lying in the interval $[-1, 1]$ and formulate the unit vector $\mathbf{u}$ as

$$\mathbf{u} = \frac{1}{(r_1^2 + r_2^2 + \cdots + r_n^2)^{1/2}} \begin{Bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{Bmatrix} \tag{6.19}$$

The directions generated using Eq. (6.19) are expected to have a bias toward the diagonals of the unit hypercube [6.3]. To avoid such a bias, the length of the vector, R, is computed as

$$R = (r_1^2 + r_2^2 + \cdots + r_n^2)^{1/2}$$

and the random numbers generated $(r_1, r_2, \ldots, r_n)$ are accepted only if $R \le 1$ but are discarded if $R > 1$. If the random numbers are accepted, the unbiased random vector $\mathbf{u}_i$ is given by Eq. (6.19).

5. Compute the new vector and the corresponding function value as $\mathbf{X} = \mathbf{X}_1 + \lambda\mathbf{u}$ and $f = f(\mathbf{X})$.
6. Compare the values of f and f_1. If $f < f_1$, set the new values as $\mathbf{X}_1 = \mathbf{X}$ and $f_1 = f$ and go to step 3. If $f \geq f_1$, go to step 7.
7. If $i \leq N$, set the new iteration number as $i = i + 1$ and go to step 4. On the other hand, if $i > N$, go to step 8.
8. Compute the new, reduced, step length as $\lambda = \lambda/2$. If the new step length is smaller than or equal to ε, go to step 9. Otherwise (i.e., if the new step length is greater than ε), go to step 4.
9. Stop the procedure by taking $\mathbf{X}_{\text{opt}} \approx \mathbf{X}_1$ and $f_{\text{opt}} \approx f_1$.

Example 6.3 Minimize $f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$ using random walk method from the point $\mathbf{X}_1 = \begin{Bmatrix} 0.0 \\ 0.0 \end{Bmatrix}$ with a starting step length of $\lambda = 1.0$. Take $\varepsilon = 0.05$ and $N = 100$.

Table 6.2 Minimization of f by Random Walk Method

Step length, λ	Number of trials required[a]	Components of $\mathbf{X}_1 + \lambda\mathbf{u}$: 1	Components of $\mathbf{X}_1 + \lambda\mathbf{u}$: 2	Current objective function value, $f_1 = f(\mathbf{X}_1 + \lambda\mathbf{u})$
1.0	1	−0.93696	0.34943	−0.06329
1.0	2	−1.15271	1.32588	−1.11986
	Next 100 trials did not reduce the function value.			
0.5	1	−1.34361	1.78800	−1.12884
0.5	3	−1.07318	1.36744	−1.20232
	Next 100 trials did not reduce the function value.			
0.25	4	−0.86419	1.23025	−1.21362
0.25	2	−0.86955	1.48019	−1.22074
0.25	8	−1.10661	1.55958	−1.23642
0.25	30	−0.94278	1.37074	−1.24154
0.25	6	−1.08729	1.57474	−1.24222
0.25	50	−0.92606	1.38368	−1.24274
0.25	23	−1.07912	1.58135	−1.24374
	Next 100 trials did not reduce the function value.			
0.125	1	−0.97986	1.50538	−1.24894
	Next 100 trials did not reduce the function value.			
0.0625	100 trials did not reduce the function value.			
0.03125	As this step length is smaller than ϵ, the program is terminated.			

[a]Out of the directions generated that satisfy $R \leq 1$, number of trials required to find a direction that also reduces the value of f.

Random Walk Method with Direction Exploitation

In the random walk method described in Section 6.2.2, we proceed to generate a new unit random vector $\mathbf{u}_{i+1}$ as soon as we find that $\mathbf{u}_i$ is successful in reducing the function value for a fixed step length λ. However, we can expect to achieve a further decrease in the function value by taking a longer step length along the direction $\mathbf{u}_i$. Thus the random walk method can be improved if the maximum possible step is taken along each successful direction. This can be achieved by using any of the one-dimensional minimization methods discussed in Chapter 5. According to this procedure, the new vector $\mathbf{X}_{i+1}$ is found as

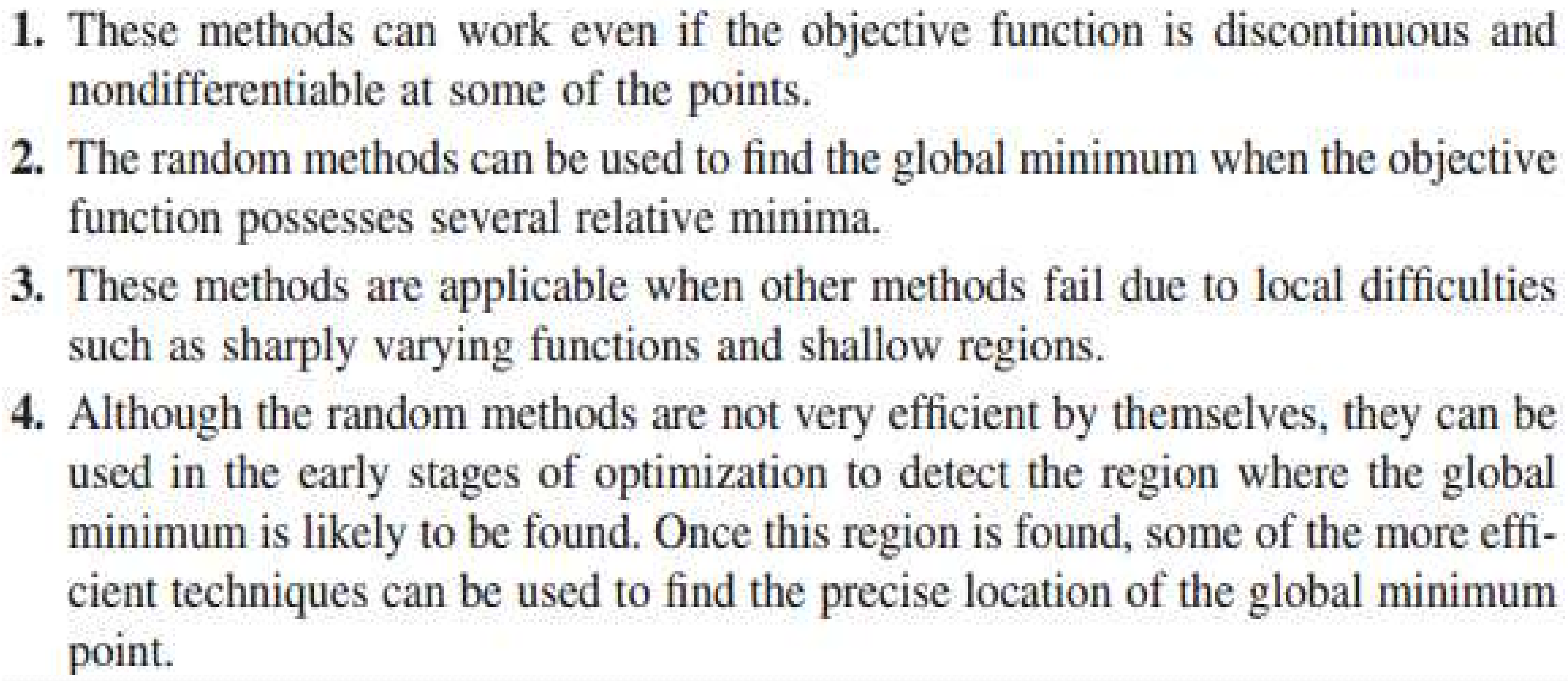
$$\mathbf{X}_{i+1} = \mathbf{X}_i + \lambda_i^* \mathbf{u}_i \tag{6.20}$$

where λ_i^* is the optimal step length found along the direction $\mathbf{u}_i$ so that

$$f_{i+1} = f(\mathbf{X}_i + \lambda_i^* \mathbf{u}_i) = \min_{\lambda_i} f(\mathbf{X}_i + \lambda_i \mathbf{u}_i) \tag{6.21}$$

The search method incorporating this feature is called the *random walk method with direction exploitation*.

Πλεονεκτήματα μεθόδων τυχαίας αναζήτησης



1. These methods can work even if the objective function is discontinuous and nondifferentiable at some of the points.
2. The random methods can be used to find the global minimum when the objective function possesses several relative minima.
3. These methods are applicable when other methods fail due to local difficulties such as sharply varying functions and shallow regions.
4. Although the random methods are not very efficient by themselves, they can be used in the early stages of optimization to detect the region where the global minimum is likely to be found. Once this region is found, some of the more efficient techniques can be used to find the precise location of the global minimum point.

Μέθοδος αναζήτησης σε κάνναβο
Grid search method

This method involves setting up a suitable grid in the design space, evaluating the objective function at all the gird points, and finding the grid point corresponding to the lowest function value. For example, if the lower and upper bounds on the ith design variable are known to be l_i and u_i, respectively, we can divide the range (l_i, u_i) into $p_i - 1$ equal parts so that $x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(pi)}$ denote the grid points along the x_i axis $(i = 1, 2, \ldots, n)$. This leads to a total of $p_1 p_2 \cdots p_n$ grid points in the design space. A grid with $p_i = 4$ is shown in a two-dimensional design space in Fig. 6.4. The grid points can also be chosen based on methods of experimental design [6.4, 6.5]. It can be seen that the grid method requires prohibitively large number of function evaluations in most practical problems. For example, for a problem with 10 design variables ($n = 10$), the number of grid points will be $3^{10} = 59{,}049$ with $p_i = 3$ and $4^{10} = 1{,}048{,}576$ with $p_i = 4$. However, for problems with a small number of design variables, the grid method can be used conveniently to find an approximate minimum. Also, the grid method can be used to find a good starting point for one of the more efficient methods.

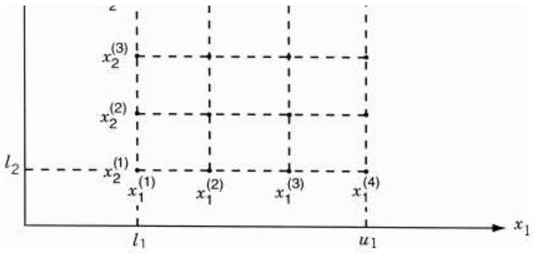
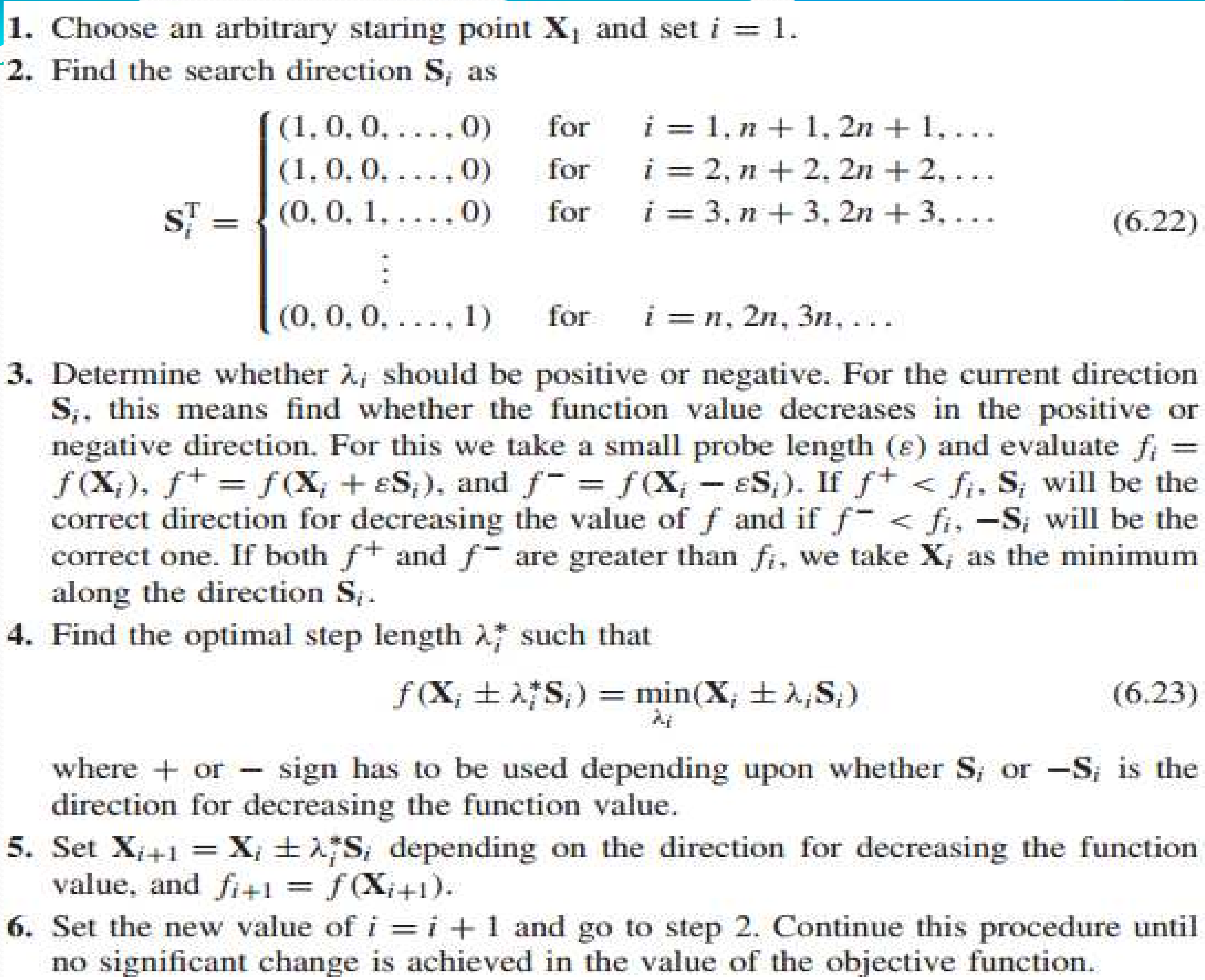


Figure 6.4 Grid with $p_i = 4$.

Univariate method

- Στη συγκεκριμένη μέθοδο αλλάζει μία μεταβλητή κάθε φορά και επιδιώκεται η διαδοχική προσέγγιση του βέλτιστου σημείου.
- Ξεκινώντας από ένα βασικό σημείο $\mathbf{X_i}$ στην *i-στη* επανάληψη μένουν σταθερές οι n-1 μεταβλητές και αλλάζει μόνο η απομένουσα μεταβλητή.
- Επομένως το πρόβλημα γίνεται μονοδιάστατο και εφαρμόζεται οποιαδήποτε τεχνική της προηγούμενης διάλεξης, ώστε να παραχθεί το επόμενο σημείο $\mathbf{X_{i+1}}$.
- Η νέα κατεύθυνση καθορίζεται τροποποιώντας οποιαδήποτε από τις απομένουσες n-1 μεταβλητές που ήταν σταθερές στο προηγούμενο βήμα. Όταν ελεγχθούν όλες οι κατευθύνσεις ο κύκλος ολοκληρώνεται και ξεκινάει η διαδικασία ξανά από την αρχή έως ότου να μην επέρχεται βελτίωση για καμία κατεύθυνση.

Univariate method - διαδικασία



1. Choose an arbitrary staring point $\mathbf{X}_1$ and set $i = 1$.
2. Find the search direction $\mathbf{S}_i$ as

$$\mathbf{S}_i^{\mathrm{T}} = \begin{cases} (1, 0, 0, \ldots, 0) & \text{for} \quad i = 1, n+1, 2n+1, \ldots \\ (1, 0, 0, \ldots, 0) & \text{for} \quad i = 2, n+2, 2n+2, \ldots \\ (0, 0, 1, \ldots, 0) & \text{for} \quad i = 3, n+3, 2n+3, \ldots \\ \quad\vdots & \\ (0, 0, 0, \ldots, 1) & \text{for} \quad i = n, 2n, 3n, \ldots \end{cases} \tag{6.22}$$

3. Determine whether λ_i should be positive or negative. For the current direction $\mathbf{S}_i$, this means find whether the function value decreases in the positive or negative direction. For this we take a small probe length (ε) and evaluate $f_i = f(\mathbf{X}_i)$, $f^+ = f(\mathbf{X}_i + \varepsilon\mathbf{S}_i)$, and $f^- = f(\mathbf{X}_i - \varepsilon\mathbf{S}_i)$. If $f^+ < f_i$, $\mathbf{S}_i$ will be the correct direction for decreasing the value of f and if $f^- < f_i$, $-\mathbf{S}_i$ will be the correct one. If both f^+ and f^- are greater than f_i, we take $\mathbf{X}_i$ as the minimum along the direction $\mathbf{S}_i$.
4. Find the optimal step length λ_i^* such that

$$f(\mathbf{X}_i \pm \lambda_i^* \mathbf{S}_i) = \min_{\lambda_i}(\mathbf{X}_i \pm \lambda_i \mathbf{S}_i) \tag{6.23}$$

   where $+$ or $-$ sign has to be used depending upon whether $\mathbf{S}_i$ or $-\mathbf{S}_i$ is the direction for decreasing the function value.
5. Set $\mathbf{X}_{i+1} = \mathbf{X}_i \pm \lambda_i^* \mathbf{S}_i$ depending on the direction for decreasing the function value, and $f_{i+1} = f(\mathbf{X}_{i+1})$.
6. Set the new value of $i = i + 1$ and go to step 2. Continue this procedure until no significant change is achieved in the value of the objective function.

The univariate method is very simple and can be implemented easily. However, it will not converge rapidly to the optimum solution, as it has a tendency to oscillate with steadily decreasing progress toward the optimum. Hence it will be better to stop the computations at some point near to the optimum point rather than trying to find the precise optimum point. In theory, the univariate method can be applied to find the minimum of any function that possesses continuous derivatives. However, if the function has a steep valley, the method may not even converge. For example, consider the contours of a function of two variables with a valley as shown in Fig. 6.5. If the univariate search starts at point P, the function value cannot be decreased either in the direction $\pm\mathbf{S}_1$ or in the direction $\pm\mathbf{S}_2$. Thus the search comes to a halt and one may be misled to take the point P, which is certainly not the optimum point, as the optimum point. This situation arises whenever the value of the probe length ε needed for detecting the proper direction ($\pm\mathbf{S}_1$ or $\pm\mathbf{S}_2$) happens to be less than the number of significant figures used in the computations

Univariate method

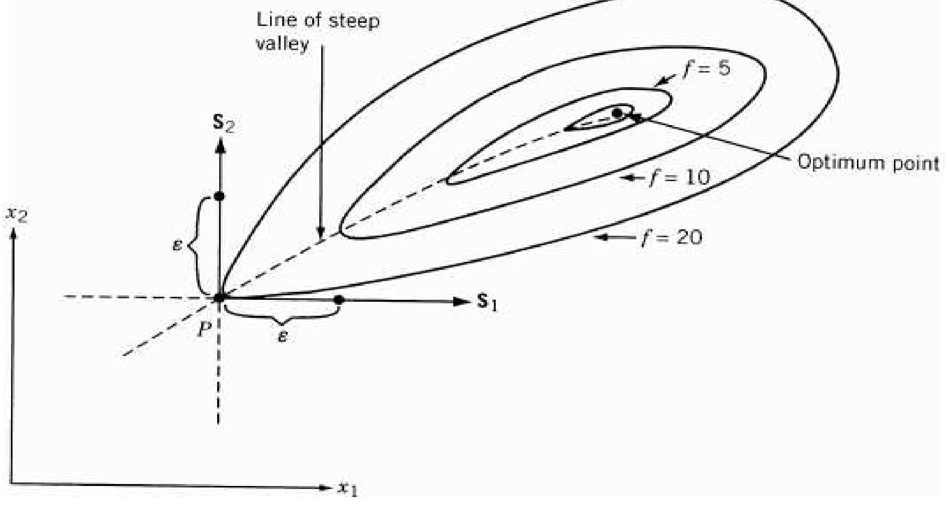


Figure 6.5 Failure of the univariate method on a steep valley.

Example 6.4 Minimize $f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$ with the starting point (0, 0).

SOLUTION We will take the probe length (ε) as 0.01 to find the correct direction for decreasing the function value in step 3. Further, we will use the differential calculus method to find the optimum step length λ_i^* along the direction $\pm\mathbf{S}_i$ in step 4.

Iteration $i = 1$

Step 2: Choose the search direction $\mathbf{S}_1$ as $\mathbf{S}_1 = \left\{\begin{smallmatrix}1\\0\end{smallmatrix}\right\}$.

Step 3: To find whether the value of f decreases along $\mathbf{S}_1$ or $-\mathbf{S}_1$, we use the probe length ε. Since

$$f_1 = f(\mathbf{X}_1) = f(0,0) = 0,$$

$$f^+ = f(\mathbf{X}_1 + \varepsilon\mathbf{S}_1) = f(\varepsilon, 0) = 0.01 - 0 + 2(0.0001)$$
$$+ 0 + 0 = 0.0102 > f_1$$

$$f^- = f(\mathbf{X}_1 - \varepsilon\mathbf{S}_1) = f(-\varepsilon, 0) = -0.01 - 0 + 2(0.0001)$$
$$+ 0 + 0 = -0.0098 < f_1,$$

$-\mathbf{S}_1$ is the correct direction for minimizing f from $\mathbf{X}_1$.

Step 4: To find the optimum step length λ_1^*, we minimize

$$f(\mathbf{X}_1 - \lambda_1\mathbf{S}_1) = f(-\lambda_1, 0)$$
$$= (-\lambda_1) - 0 + 2(-\lambda_1)^2 + 0 + 0 = 2\lambda_1^2 - \lambda_1$$

As $df/d\lambda_1 = 0$ at $\lambda_1 = \frac{1}{4}$, we have $\lambda_1^* = \frac{1}{4}$.

Step 5: Set

$$\mathbf{X}_2 = \mathbf{X}_1 - \lambda_1^*\mathbf{S}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} - \frac{1}{4}\begin{Bmatrix} 1 \\ 0 \end{Bmatrix} = \begin{Bmatrix} -\frac{1}{4} \\ 0 \end{Bmatrix}$$

$$f_2 = f(\mathbf{X}_2) = f(-\tfrac{1}{4}, 0) = -\tfrac{1}{8}.$$

***Iteration* $i = 2$**

Step 2: Choose the search direction $\mathbf{S}_2$ as $\mathbf{S}_2 = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix}$.

Step 3: Since $f_2 = f(\mathbf{X}_2) = -0.125$,

$$f^+ = f(\mathbf{X}_2 + \varepsilon \mathbf{S}_2) = f(-0.25, 0.01) = -0.1399 < f_2$$

$$f^- = f(\mathbf{X}_2 + \varepsilon \mathbf{S}_2) = f(-0.25, -0.01) = -0.1099 > f_2$$

$\mathbf{S}_2$ is the correct direction for decreasing the value of f from $\mathbf{X}_2$.

Step 4: We minimize $f(\mathbf{X}_2 + \lambda_2 \mathbf{S}_2)$ to find λ_2^*.

Here

$$\begin{aligned} f(\mathbf{X}_2 + \lambda_2 \mathbf{S}_2) &= f(-0.25, \lambda_2) \\ &= -0.25 - \lambda_2 + 2(0.25)^2 - 2(0.25)(\lambda_2) + \lambda_2^2 \\ &= \lambda_2^2 - 1.5\lambda_2 - 0.125 \end{aligned}$$

$$\frac{df}{d\lambda_2} = 2\lambda_2 - 1.5 = 0 \quad \text{at} \quad \lambda_2^* = 0.75$$

Step 5: Set

$$\mathbf{X}_3 = \mathbf{X}_2 + \lambda_2^* \mathbf{S}_2 = \begin{Bmatrix} -0.25 \\ 0 \end{Bmatrix} + 0.75 \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} = \begin{Bmatrix} -0.25 \\ 0.75 \end{Bmatrix}$$

$$f_3 = f(\mathbf{X}_3) = -0.6875$$

Next we set the iteration number as $i = 3$, and continue the procedure until the optimum solution $\mathbf{X}^* = \begin{Bmatrix} -1.0 \\ 1.5 \end{Bmatrix}$ with $f(\mathbf{X}^*) = -1.25$ is found.

Note: If the method is to be computerized, a suitable convergence criterion has to be used to test the point $\mathbf{X}_{i+1} (i = 1, 2, \ldots)$ for optimality.

Αναζήτηση σε μοτίβο – Pattern search

- Στη μέθοδο univariate αναζητείται το βέλτιστο σε κατευθύνσεις παράλληλα στους άξονες των μεταβλητών.
- Αλλάζοντας κατάλληλα την κατεύθυνση αναζήτησης μπορεί να επιταχυνθεί η διαδικασία.

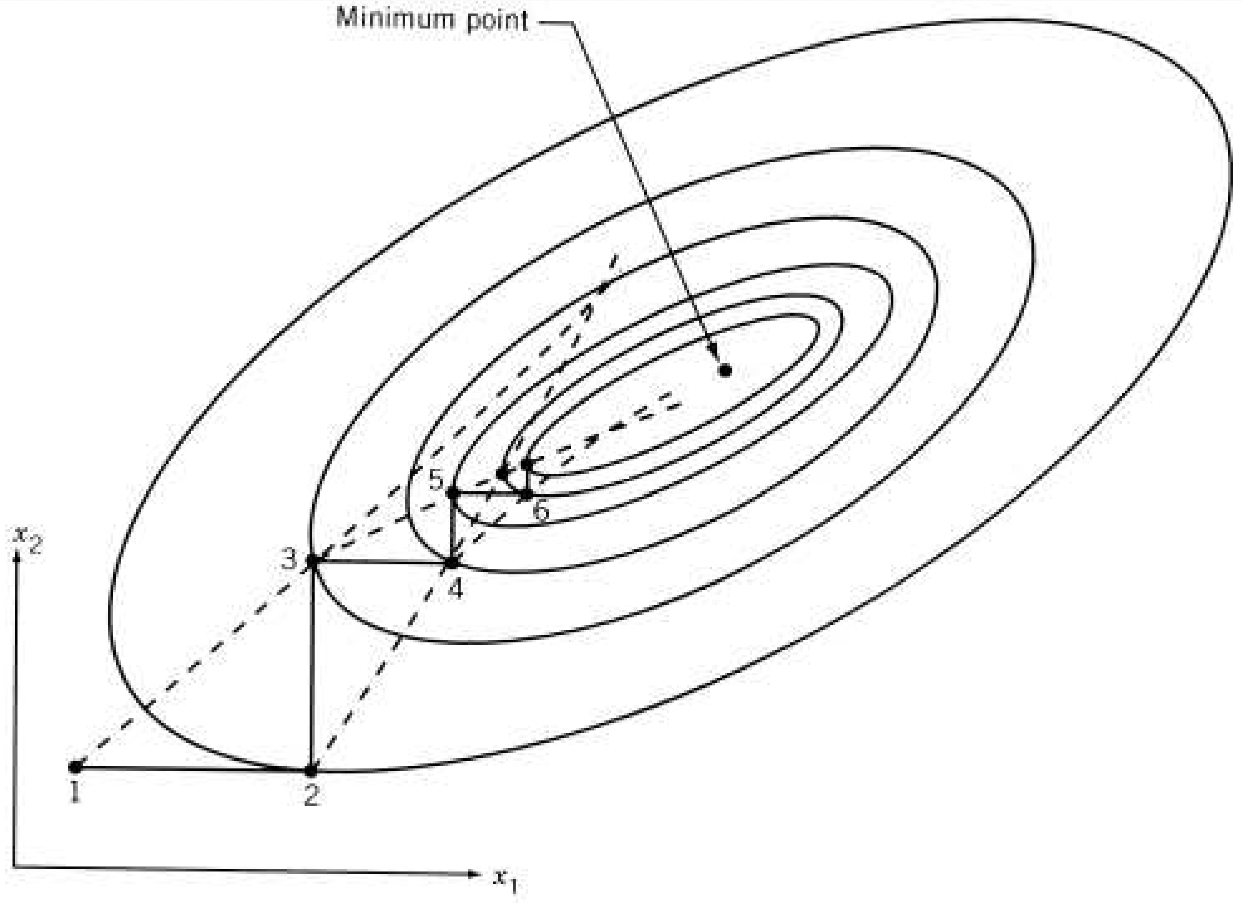


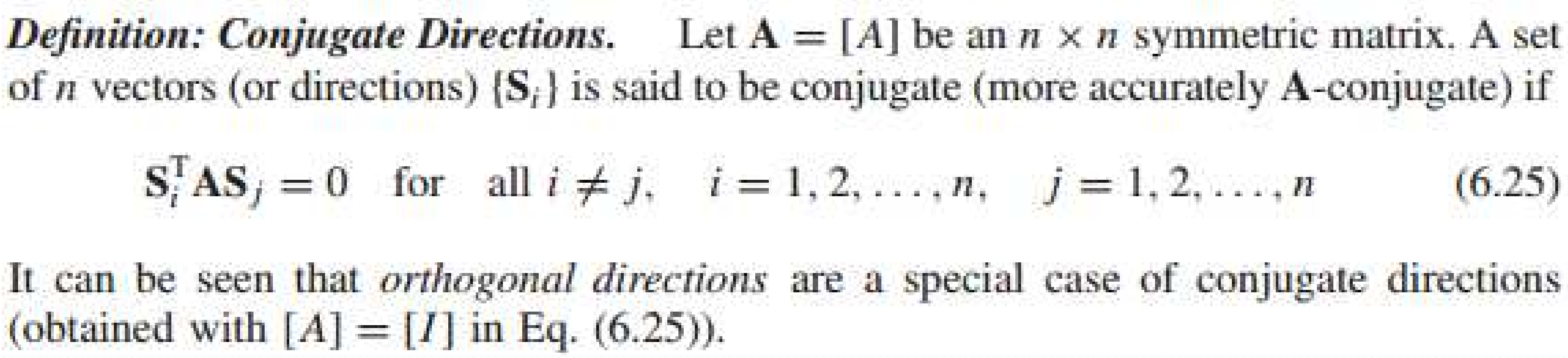
Figure 6.6 Lines defined by the alternate points lie in the general direction of the minimum.

Αναζήτηση σε μοτίβο – Pattern search

- Γενικά η αναζήτηση σε μοτίβο υλοποιεί *n* βήματα μεταβάλλοντας μόνο μια μεταβλητή κάθε φορά, όπου *n* ο αριθμός των μεταβλητών.
- Έπειτα αναζητά το βέλτιστο στην κατεύθυνση $\mathbf{S_i}$, η οποία καθορίζεται: $\mathbf{S_i = X_i - X_{i-n}}$
- όπου $\mathbf{X_i}$ το σημείο που βρέθηκε στο τέλος των *n* βημάτων και $\mathbf{X_{i-n}}$ το αρχικό σημείο του κύκλου αναζήτησης.

Μέθοδος του Powell

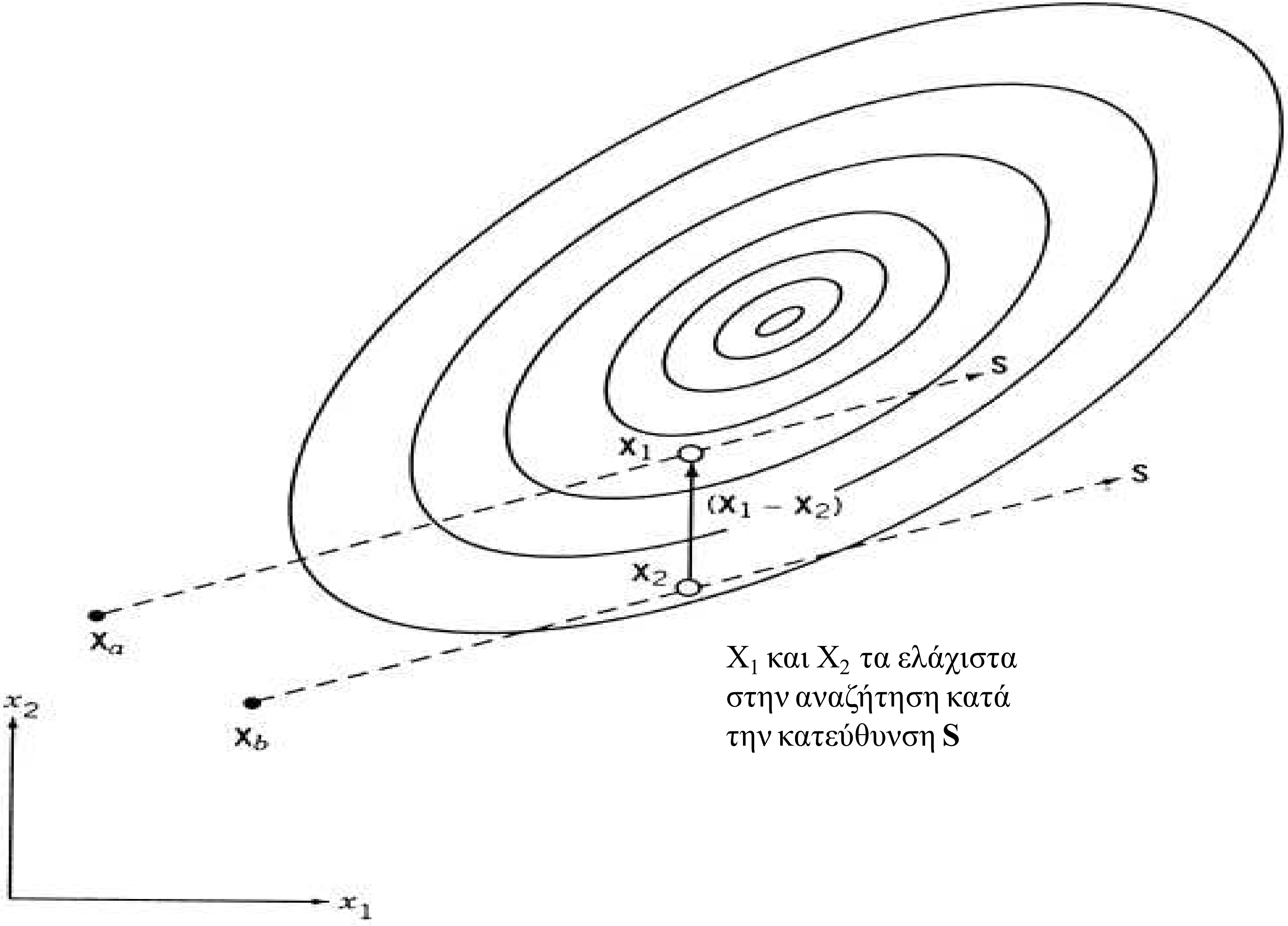
- Αποτελεί επέκταση της βασικής αναζήτησης σε μοτίβο.
- Είναι αρκετά διαδεδομένη.
- Κάνει χρήση των «conjugate directions» για να εκτιμήσει την κατεύθυνση αναζήτησης:



Definition: Conjugate Directions. Let $\mathbf{A} = [A]$ be an $n \times n$ symmetric matrix. A set of n vectors (or directions) $\{\mathbf{S}_i\}$ is said to be conjugate (more accurately $\mathbf{A}$-conjugate) if

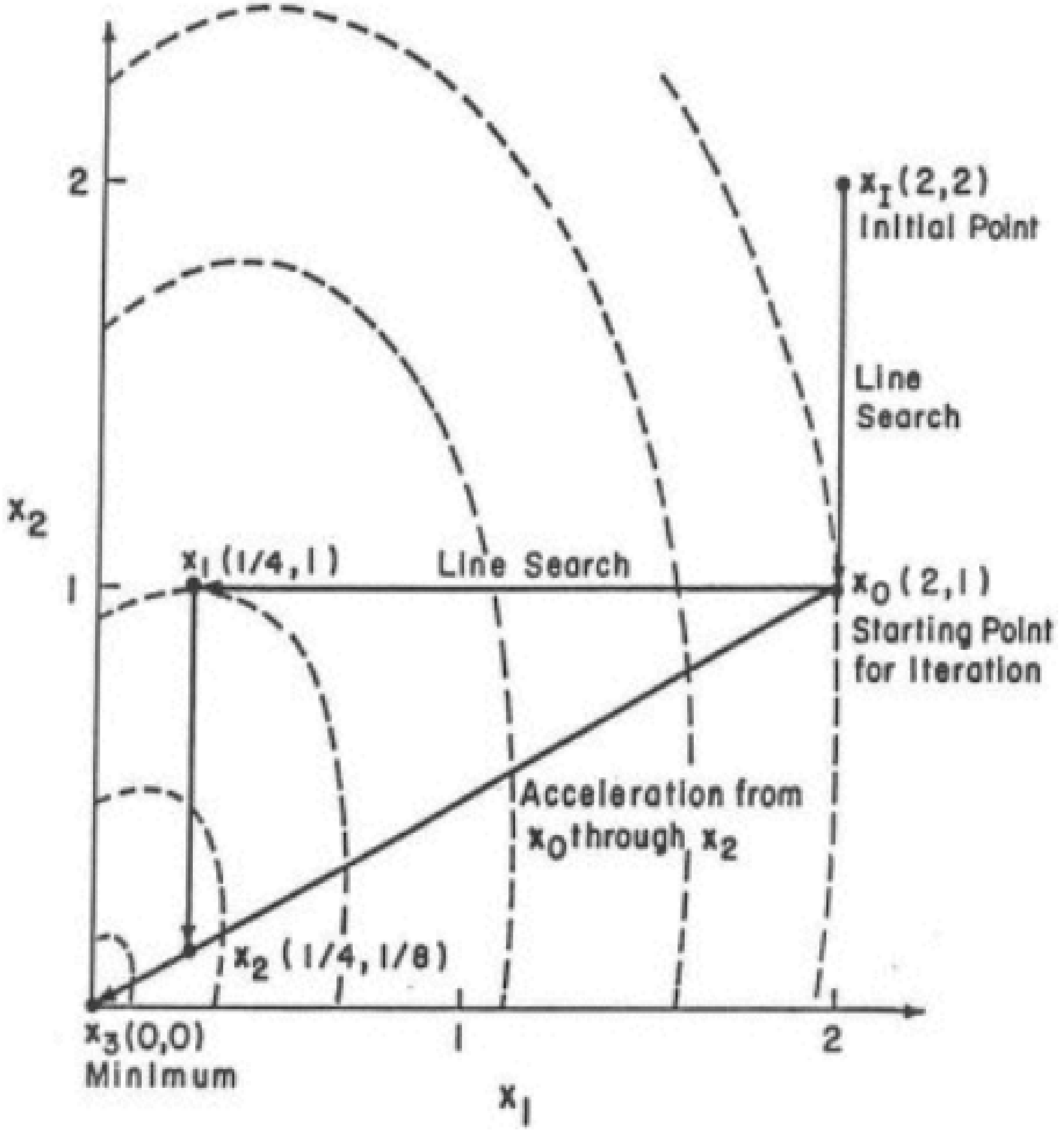
$$\mathbf{S}_i^{\mathrm{T}} \mathbf{A} \mathbf{S}_j = 0 \quad \text{for} \quad \text{all } i \neq j, \quad i = 1, 2, \ldots, n, \quad j = 1, 2, \ldots, n \tag{6.25}$$

It can be seen that *orthogonal directions* are a special case of conjugate directions (obtained with $[A] = [I]$ in Eq. (6.25)).



X_1 και X_2 τα ελάχιστα στην αναζήτηση κατά την κατεύθυνση **S**

Figure 6.7 Conjugate directions.



$x_I(2,2)$ Initial Point
Line Search
$x_0(2,1)$ Starting Point for Iteration
$x_1(1/4,1)$
Line Search
Acceleration from x_0 through x_2
$x_2(1/4,1/8)$
$x_3(0,0)$ Minimum
x_2
x_1
2
1
1
2

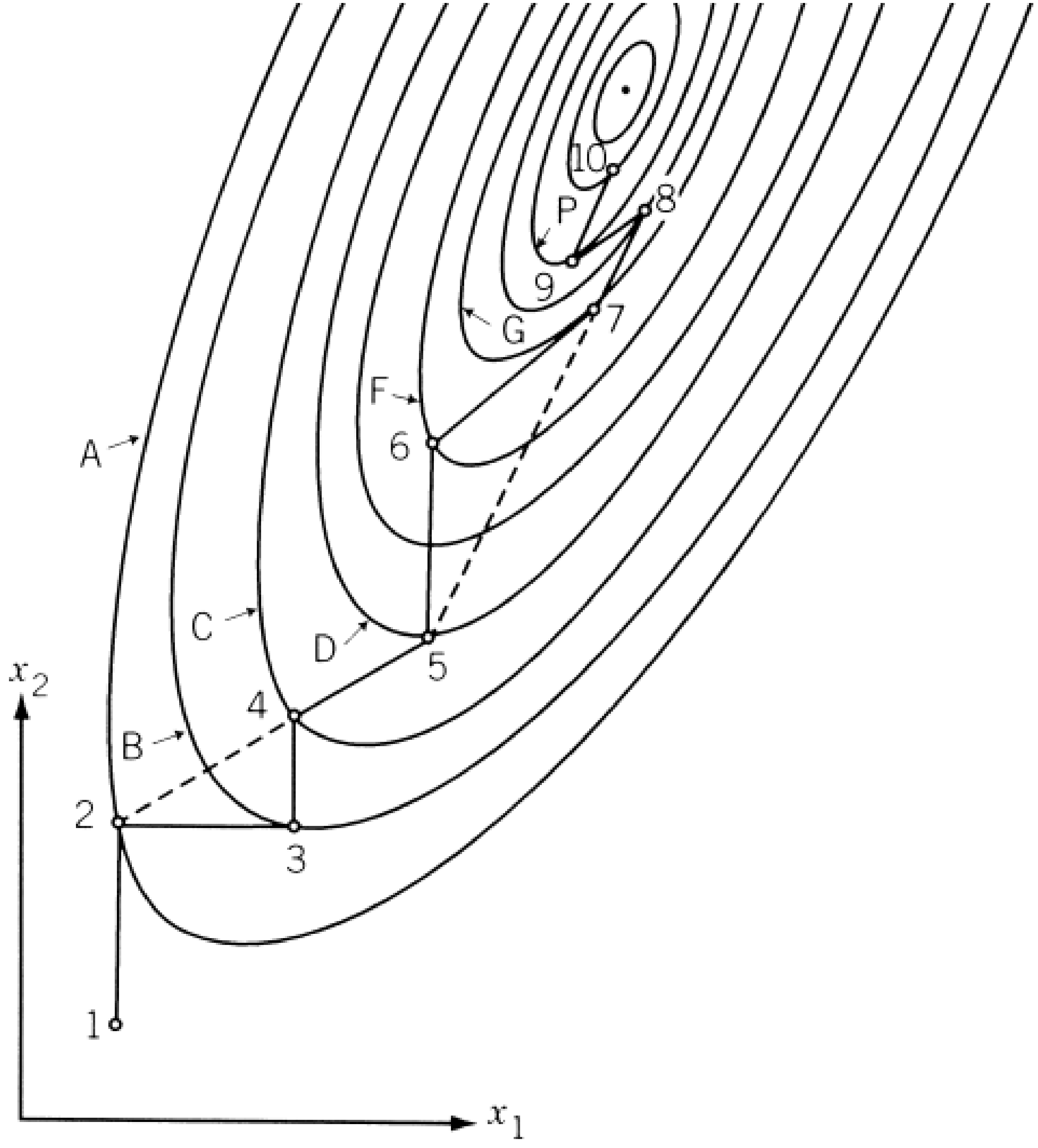
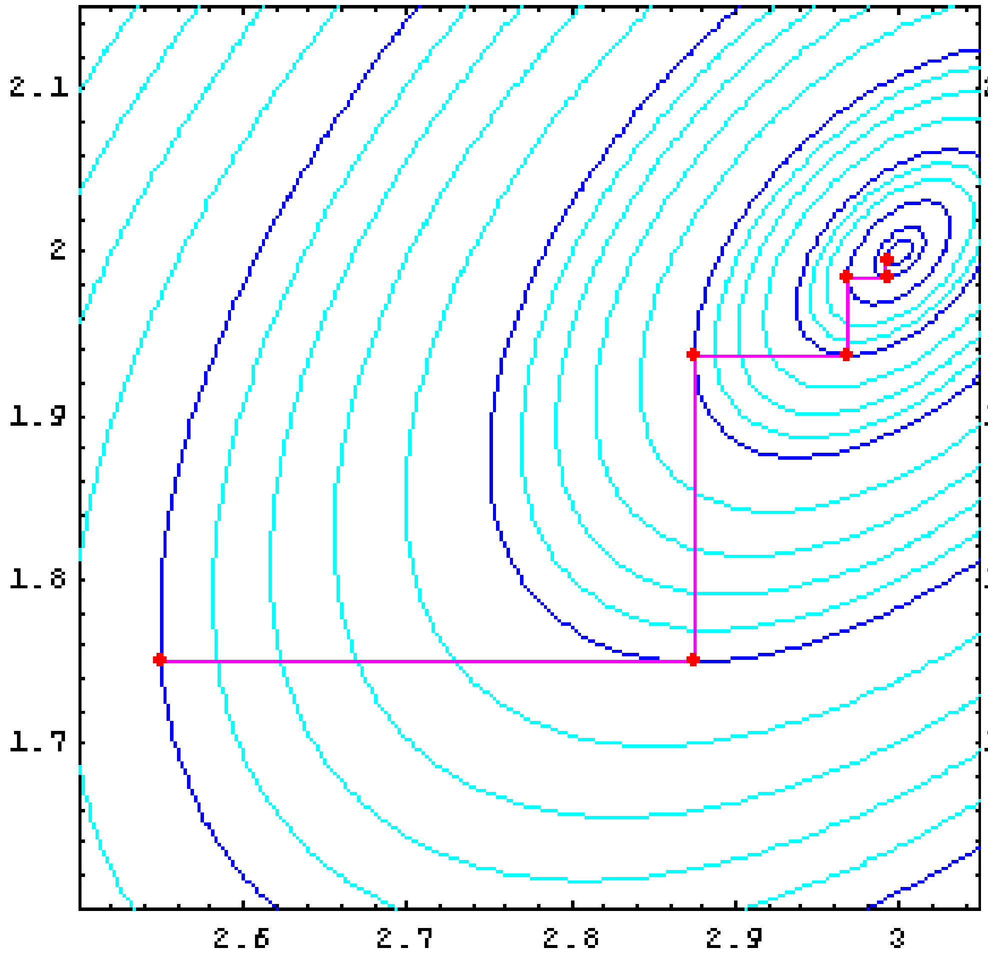
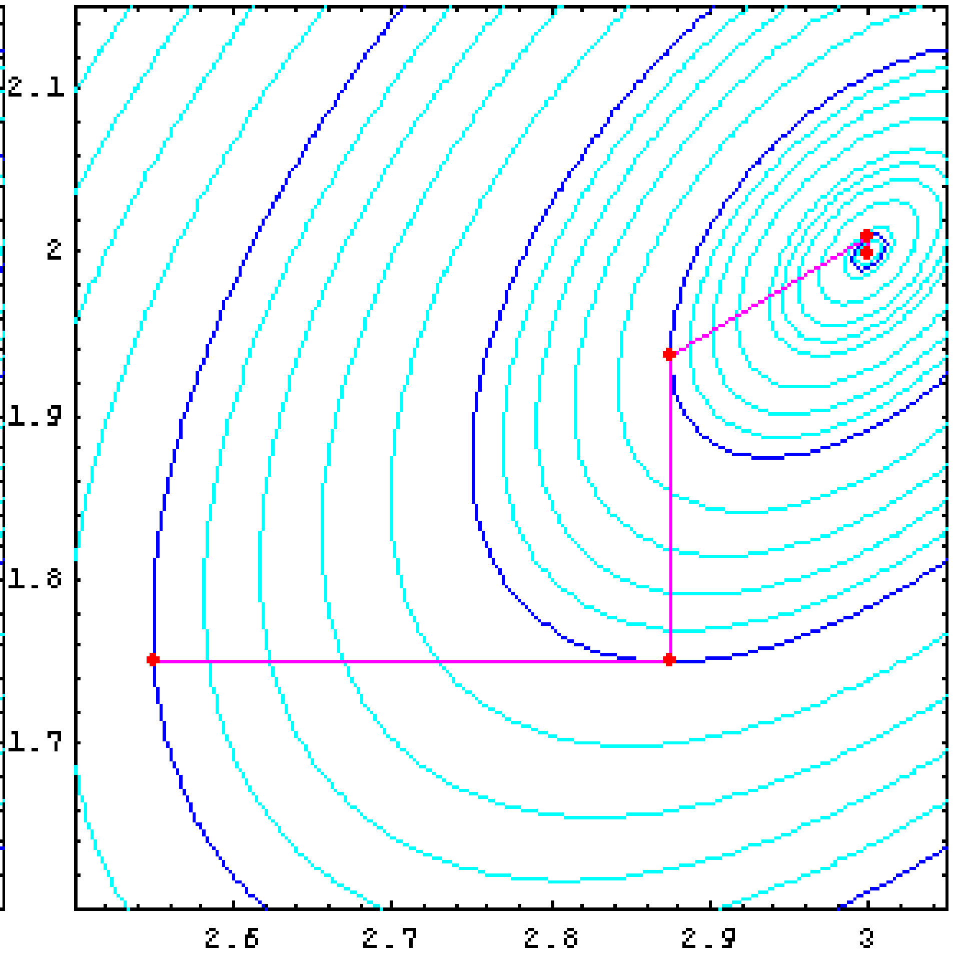
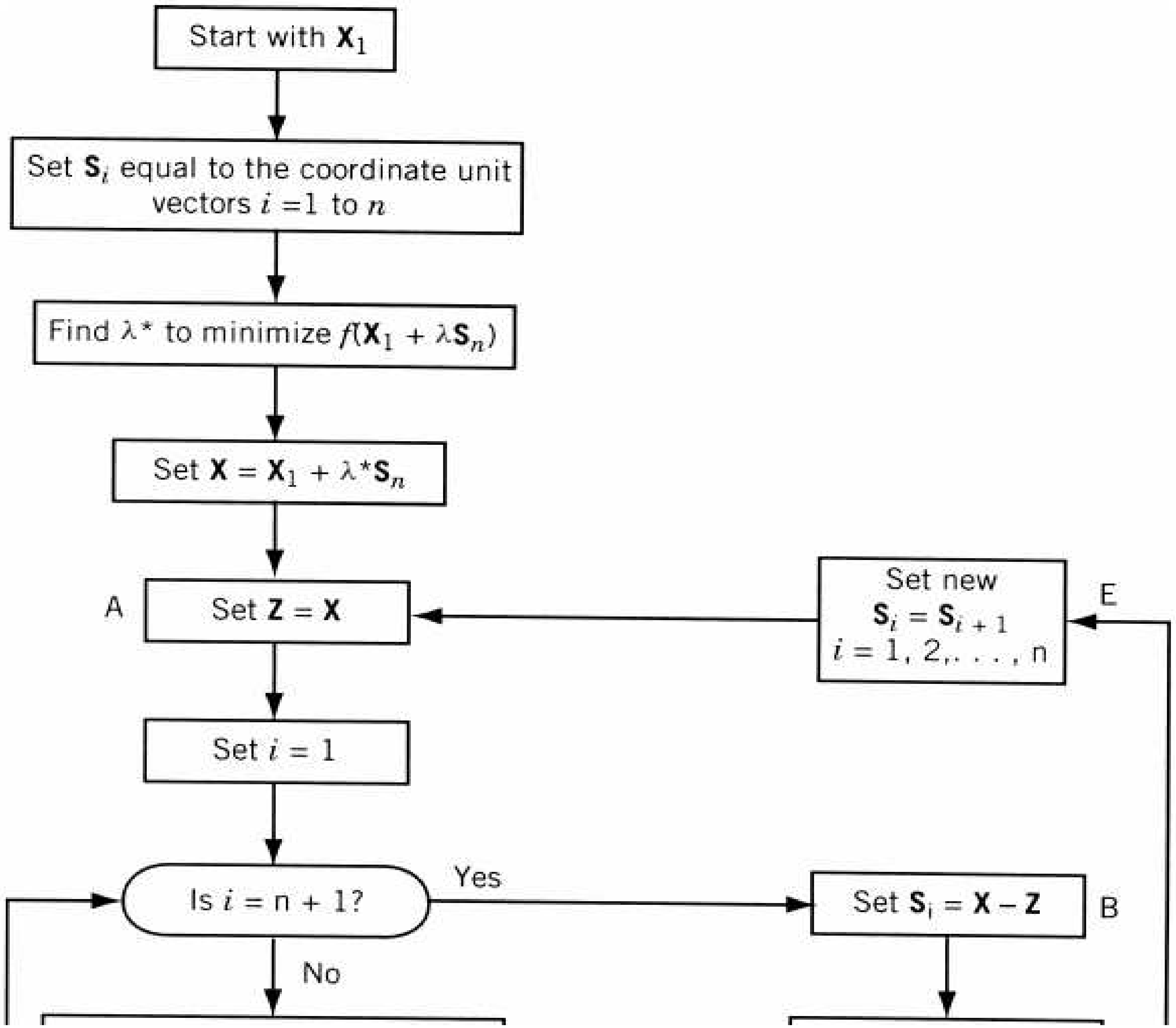


Figure 6.8 Progress of Powell's method.

Descent method vs Powell's







Start with $\mathbf{X}_1$
Set $\mathbf{S}_i$ equal to the coordinate unit vectors $i = 1$ to n
Find λ^* to minimize $f(\mathbf{X}_1 + \lambda \mathbf{S}_n)$
Set $\mathbf{X} = \mathbf{X}_1 + \lambda^* \mathbf{S}_n$
A
Set $\mathbf{Z} = \mathbf{X}$
Set new $\mathbf{S}_i = \mathbf{S}_{i+1}$ $i = 1, 2, \ldots, n$
E
Set $i = 1$
Is $i = n + 1$?
Yes
Set $\mathbf{S}_i = \mathbf{X} - \mathbf{Z}$
B
No

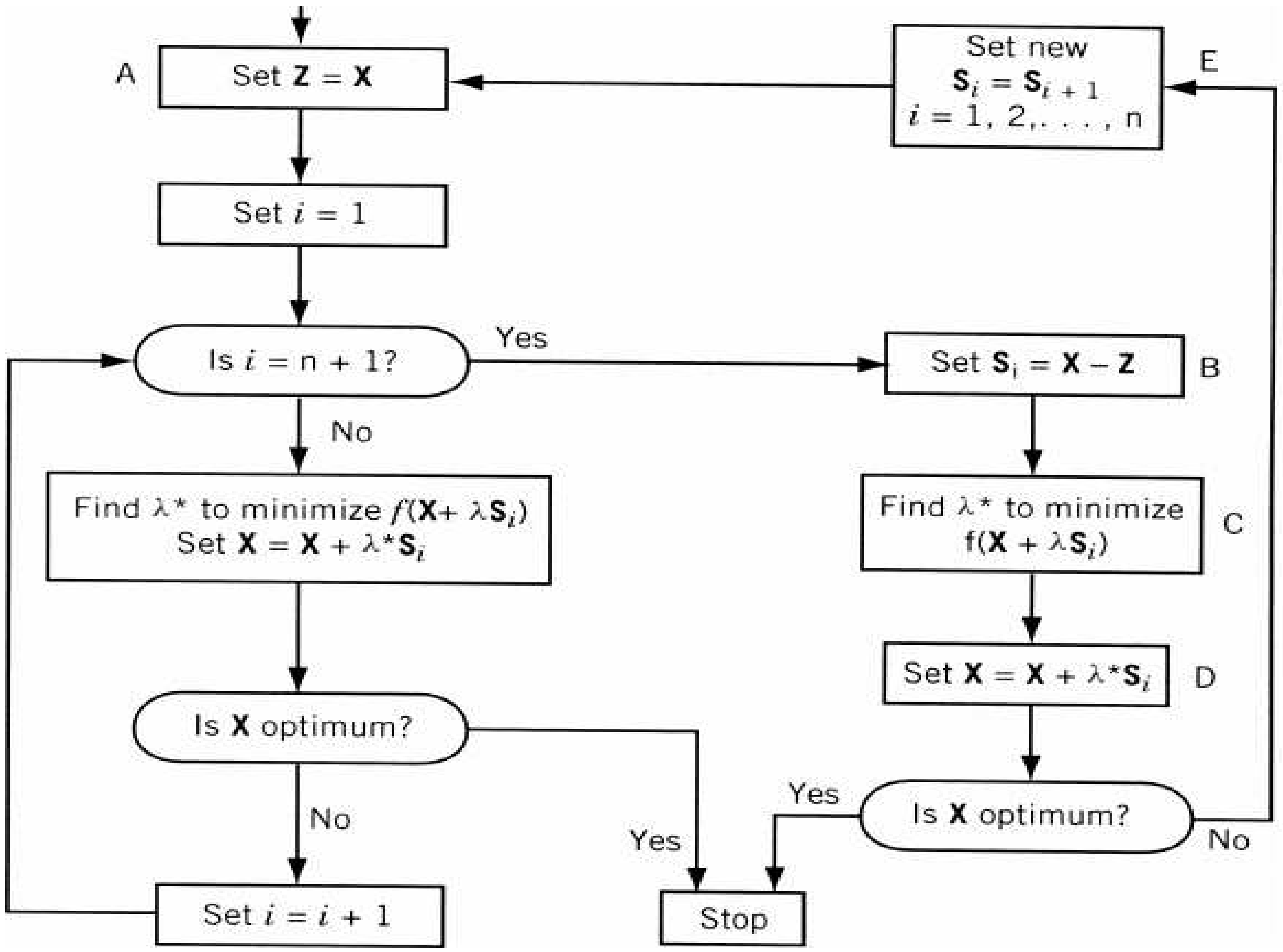
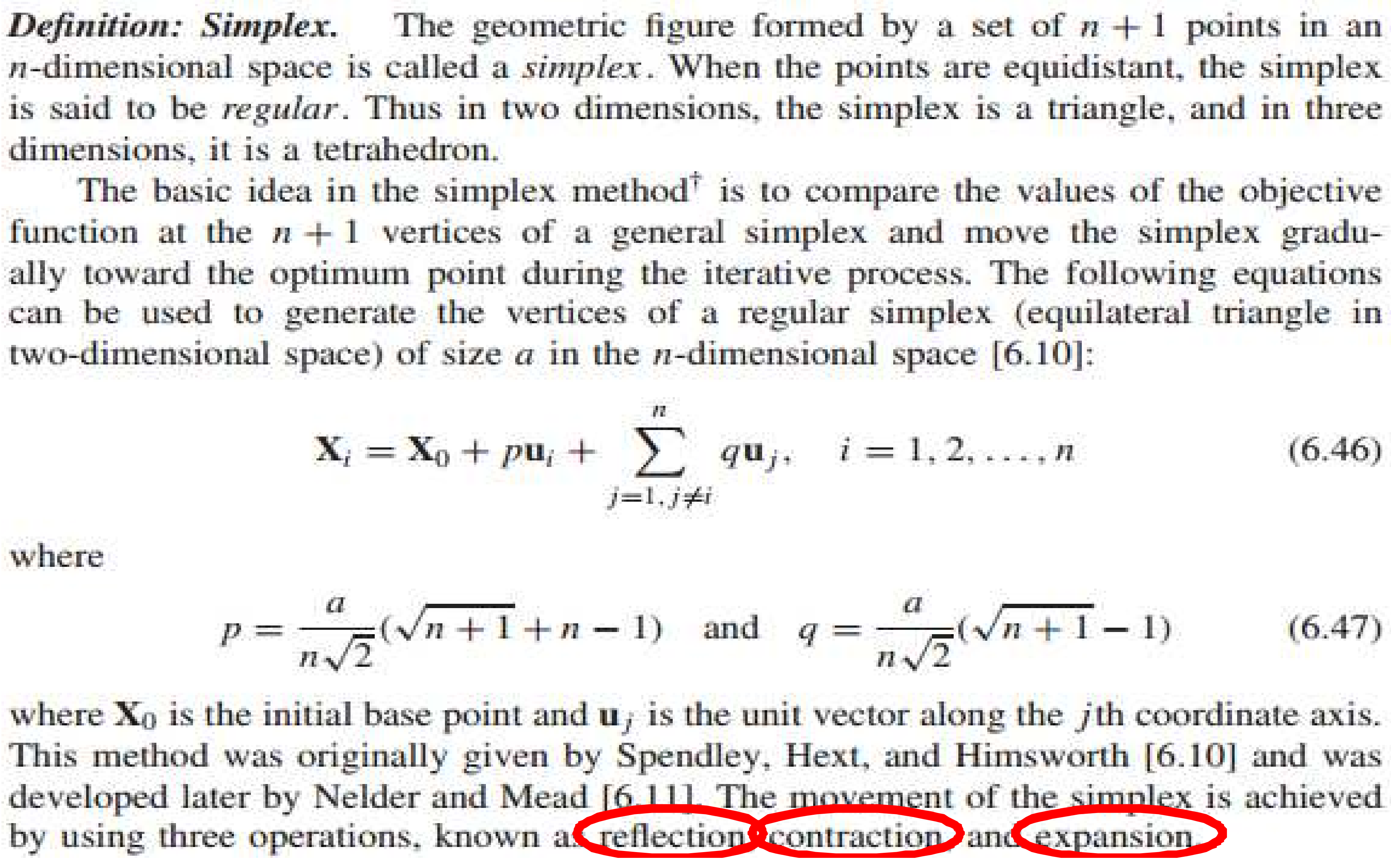

A
Set $\mathbf{Z} = \mathbf{X}$
Set $i = 1$
Is $i = n + 1$?
Yes
No
Find λ^* to minimize $f(\mathbf{X} + \lambda \mathbf{S}_i)$
Set $\mathbf{X} = \mathbf{X} + \lambda^* \mathbf{S}_i$
Is $\mathbf{X}$ optimum?
No
Yes
Set $i = i + 1$
Stop
Set new $\mathbf{S}_i = \mathbf{S}_{i+1}$, $i = 1, 2, \ldots, n$
E
Set $\mathbf{S}_i = \mathbf{X} - \mathbf{Z}$
B
Find λ^* to minimize $f(\mathbf{X} + \lambda \mathbf{S}_i)$
C
Set $\mathbf{X} = \mathbf{X} + \lambda^* \mathbf{S}_i$
D
Is $\mathbf{X}$ optimum?
Yes
No

Figure 6.9 Flowchart for Powell's Method.

Μέθοδος Simplex

- ΠΡΟΣΟΧΗ!!! ΔΕΝ ΕΧΕΙ ΣΧΕΣΗ ΜΕ ΤΗ ΜΕΘΟΔΟ SIMPLEX ΕΠΙΛΥΣΗΣ ΠΡΟΒΛΗΜΑΤΩΝ ΓΡΑΜΜΙΚΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ!!!



Definition: Simplex. The geometric figure formed by a set of $n + 1$ points in an n-dimensional space is called a *simplex*. When the points are equidistant, the simplex is said to be *regular*. Thus in two dimensions, the simplex is a triangle, and in three dimensions, it is a tetrahedron.

The basic idea in the simplex method† is to compare the values of the objective function at the $n + 1$ vertices of a general simplex and move the simplex gradually toward the optimum point during the iterative process. The following equations can be used to generate the vertices of a regular simplex (equilateral triangle in two-dimensional space) of size a in the n-dimensional space [6.10]:

$$\mathbf{X}_i = \mathbf{X}_0 + p\mathbf{u}_i + \sum_{j=1,\, j\neq i}^{n} q\mathbf{u}_j, \quad i = 1, 2, \ldots, n \tag{6.46}$$

where

$$p = \frac{a}{n\sqrt{2}}(\sqrt{n+1} + n - 1) \quad \text{and} \quad q = \frac{a}{n\sqrt{2}}(\sqrt{n+1} - 1) \tag{6.47}$$

where $\mathbf{X}_0$ is the initial base point and $\mathbf{u}_j$ is the unit vector along the jth coordinate axis. This method was originally given by Spendley, Hext, and Himsworth [6.10] and was developed later by Nelder and Mead [6.11]. The movement of the simplex is achieved by using three operations, known as reflection, contraction, and expansion.

Nelder-Mead Method

A simplex method for finding a local minimum of a function of several variables has been devised by Nelder and Mead. For two variables, a simplex is a triangle, and the method is a pattern search that compares function values at the three vertices of a triangle. The worst vertex, where $f(x, y)$ is largest, is rejected and replaced with a new vertex. A new triangle is formed and the search is continued. The process generates a sequence of triangles (which might have different shapes), for which the function values at the vertices get smaller and smaller. The size of the triangles is reduced and the coordinates of the minimum point are found.

The algorithm is stated using the term *simplex* (a generalized triangle in N dimensions) and will find the minimum of a function of N variables. It is effective and computationally compact.

Initial Triangle BGW

Let $f(x, y)$ be the function that is to be minimized. To start, we are given three vertices of a triangle: $\mathbf{V}_k = (x_k, y_k)$, $k = 1, 2, 3$. The function $f(x, y)$ is then evaluated at each of the three points: $z_k = f(x_k, y_k)$ for $k = 1, 2, 3$. The subscripts are then reordered so that $z_1 \le z_2 \le z_3$. We use the notation

$$(5) \qquad \mathbf{B} = (x_1, y_1), \qquad \mathbf{G} = (x_2, y_2), \qquad \text{and} \qquad \mathbf{W} = (x_3, y_3)$$

to help remember that $\mathbf{B}$ is the best vertex, $\mathbf{G}$ is good (next to best), and $\mathbf{W}$ is the worst vertex.

Midpoint of the Good Side

The construction process uses the midpoint of the line segment joining $\mathbf{B}$ and $\mathbf{G}$. It is found by averaging the coordinates:

$$(6) \qquad \mathbf{M} = \frac{\mathbf{B} + \mathbf{G}}{2} = \left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2} \right).$$

Reflection Using the Point $\boldsymbol{R}$

The function decreases as we move along the side of the triangle from $\boldsymbol{W}$ to $\boldsymbol{B}$, and it decreases as we move along the side from $\boldsymbol{W}$ to $\boldsymbol{G}$. Hence it is feasible that $f(x, y)$ takes on smaller values at points that lie away from $\boldsymbol{W}$ on the opposite side of the line between $\boldsymbol{B}$ and $\boldsymbol{G}$. We choose a test point $\boldsymbol{R}$ that is obtained by "reflecting" the triangle through the side $\overline{\boldsymbol{BG}}$. To determine $\boldsymbol{R}$, we first find the midpoint $\boldsymbol{M}$ of the side $\overline{\boldsymbol{BG}}$. Then draw the line segment from $\boldsymbol{W}$ to $\boldsymbol{M}$ and call its length d. This last segment is extended a distance d through $\boldsymbol{M}$ to locate the point $\boldsymbol{R}$ (see Figure 8.6). The vector formula for $\boldsymbol{R}$ is

$$\boldsymbol{R} = \boldsymbol{M} + (\boldsymbol{M} - \boldsymbol{W}) = 2\boldsymbol{M} - \boldsymbol{W}. \tag{7}$$

Expansion Using the Point $\boldsymbol{E}$

If the function value at $\boldsymbol{R}$ is smaller than the function value at $\boldsymbol{W}$, then we have moved in the correct direction toward the minimum. Perhaps the minimum is just a bit farther than the point $\boldsymbol{R}$. So we extend the line segment through $\boldsymbol{M}$ and $\boldsymbol{R}$ to the point $\boldsymbol{E}$. This forms an expanded triangle $\boldsymbol{BGE}$. The point $\boldsymbol{E}$ is found by moving an additional distance d along the line joining $\boldsymbol{M}$ and $\boldsymbol{R}$ (see Figure 8.7). If the function value at $\boldsymbol{E}$ is less than the function value at $\boldsymbol{R}$, then we have found a better vertex than $\boldsymbol{R}$. The vector formula for $\boldsymbol{E}$ is

$$\boldsymbol{E} = \boldsymbol{R} + (\boldsymbol{R} - \boldsymbol{M}) = 2\boldsymbol{R} - \boldsymbol{M}. \tag{8}$$

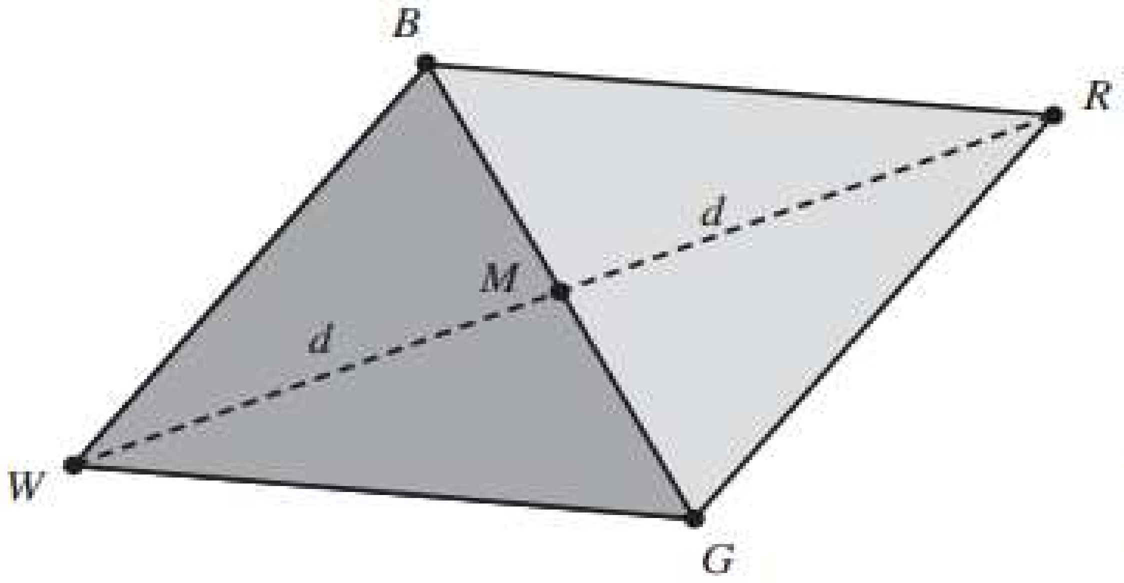


Figure 8.6 The triangle $\triangle BGW$ and midpoint $\boldsymbol{M}$ and reflected point $\boldsymbol{R}$ for the Nelder-Mead method.

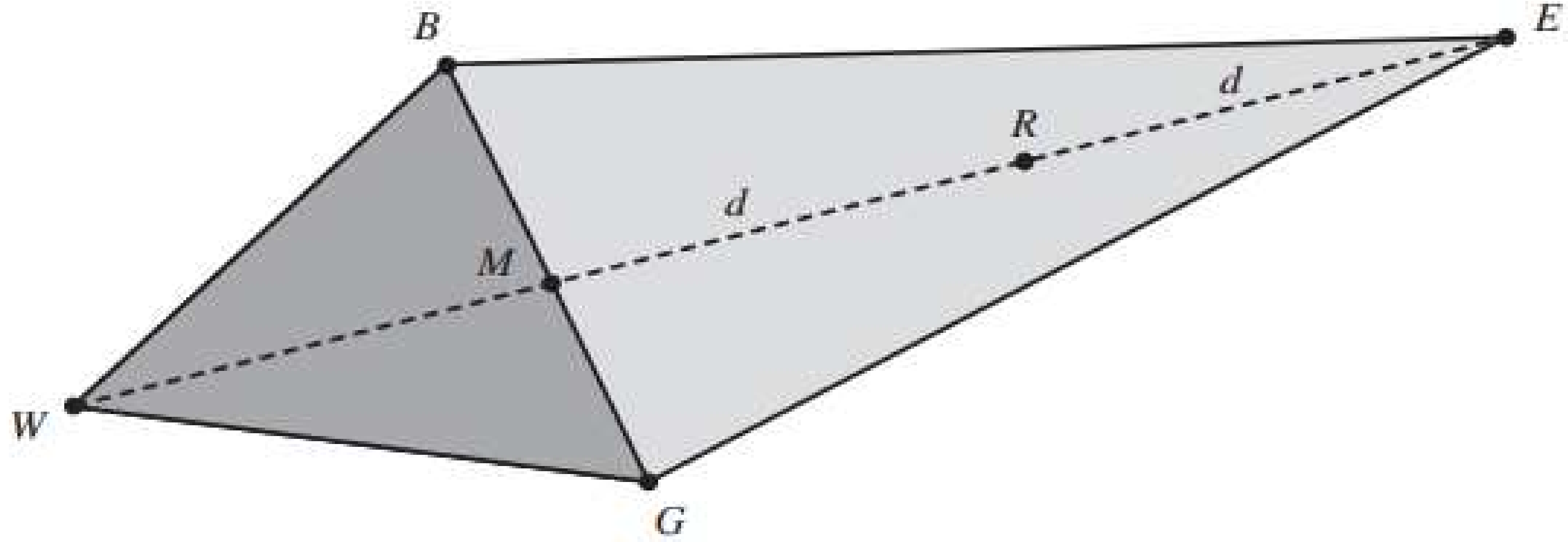


Figure 8.7 The triangle $\triangle BGW$ and point $\boldsymbol{R}$ and extended point $\boldsymbol{E}$.

Contraction Using the Point $\boldsymbol{C}$

If the function values at $\boldsymbol{R}$ and $\boldsymbol{W}$ are the same, another point must be tested. Perhaps the function is smaller at $\boldsymbol{M}$, but we cannot replace $\boldsymbol{W}$ with $\boldsymbol{M}$ because we must have a triangle. Consider the two midpoints $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ of the line segments $\overline{\boldsymbol{WM}}$ and $\overline{\boldsymbol{MR}}$, respectively (see Figure 8.8). The point with the smaller function value is called $\boldsymbol{C}$, and the new triangle is $\boldsymbol{BGC}$. *Note.* The choice between $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ might seem inappropriate for the two-dimensional case, but it is important in higher dimensions.

Shrink toward $\boldsymbol{B}$

If the function value at $\boldsymbol{C}$ is not less than the value at $\boldsymbol{W}$, the points $\boldsymbol{G}$ and $\boldsymbol{W}$ must be shrunk toward $\boldsymbol{B}$ (see Figure 8.9). The point $\boldsymbol{G}$ is replaced with $\boldsymbol{M}$, and $\boldsymbol{W}$ is replaced with $\boldsymbol{S}$, which is the midpoint of the line segment joining $\boldsymbol{B}$ with $\boldsymbol{W}$.

Logical Decisions for Each Step

A computationally efficient algorithm should perform function evaluations only if needed. In each step, a new vertex is found, which replaces $\boldsymbol{W}$. As soon as it is found, further investigation is not needed, and the iteration step is completed. The logical details for two-dimensional cases are explained in Table 8.5.

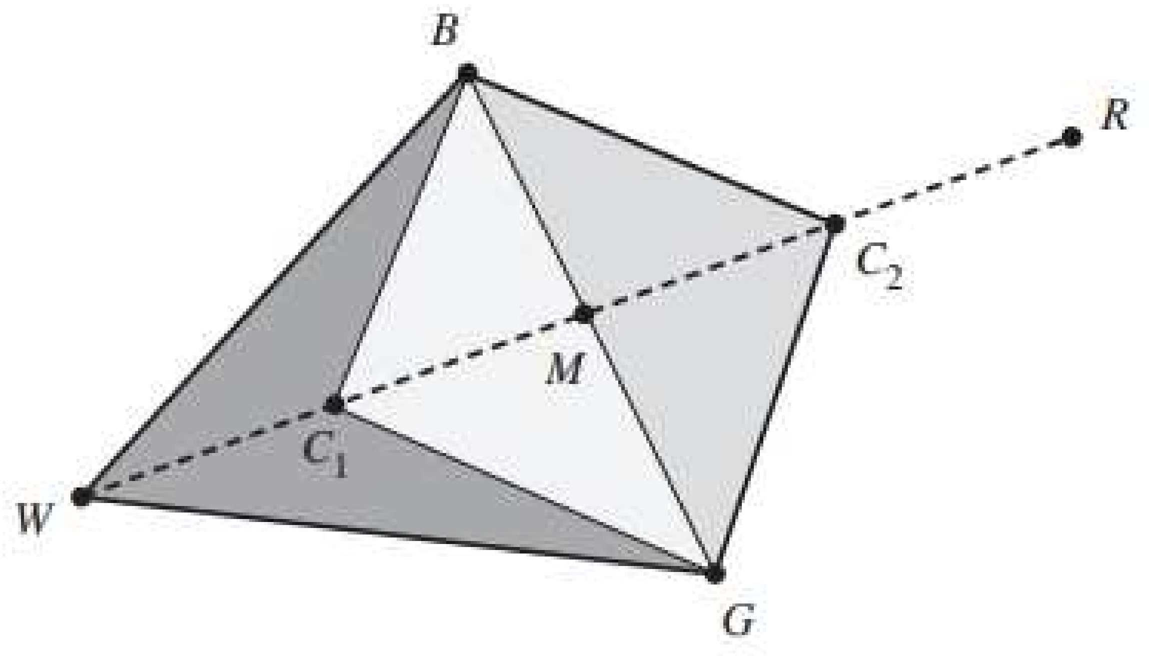


Figure 8.8 The contraction point $\boldsymbol{C}_1$ or $\boldsymbol{C}_2$ for Nelder-Mead method.

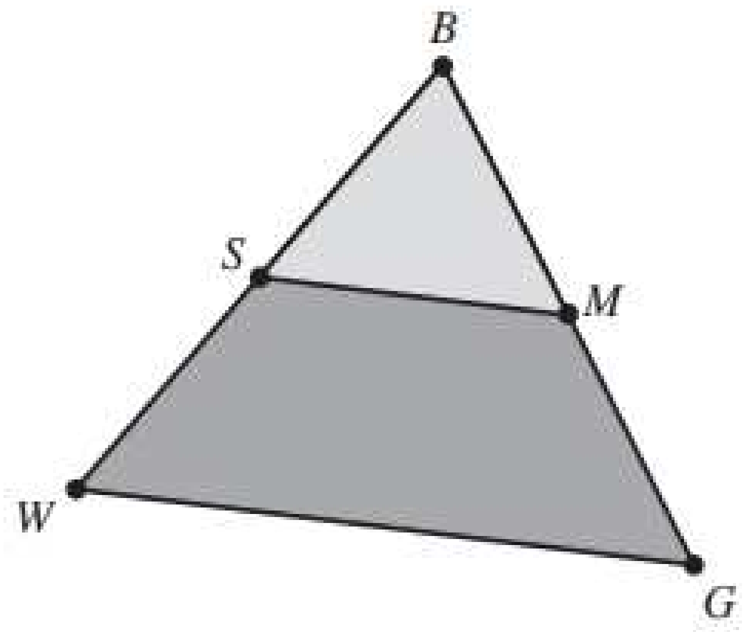


Figure 8.9 Shrinking the triangle toward $\boldsymbol{B}$.

Table 8.5 Logical Decisions for the Nelder-Mead Algorithm

```
IF $f(R) < f(G)$, THEN Perform Case (i) {either reflect or extend}
                       ELSE Perform Case (ii) {either contract or shrink}

BEGIN {Case (i).}
IF $f(B) < f(R)$ THEN
    replace $W$ with $R$
  ELSE
    Compute $E$ and $f(E)$
    IF $f(E) < f(B)$ THEN
        replace $W$ with $E$
      ELSE
        replace $W$ with $R$
    ENDIF
ENDIF
END {Case (i).}

BEGIN {Case (ii).}
IF $f(R) < f(W)$ THEN
    replace $W$ with $R$
Compute $C = (W + M)/2$
or $C = (M + R)/2$ and $f(C)$
IF $f(C) < f(W)$ THEN
    replace $W$ with $C$
  ELSE
        Compute $S$ and $f(S)$
    replace $W$ with $S$
    replace $G$ with $M$
ENDIF
END {Case (ii).}
```

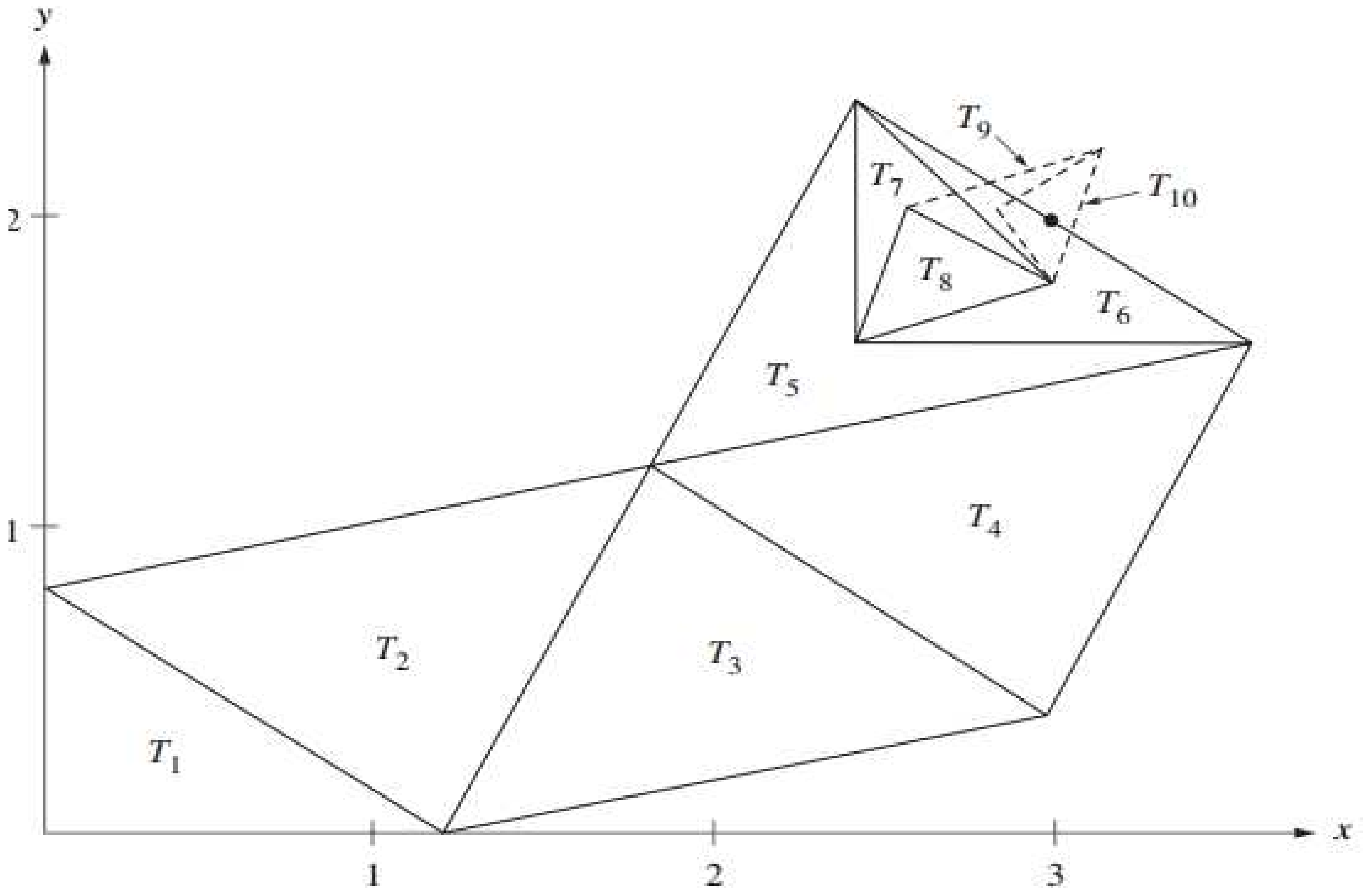
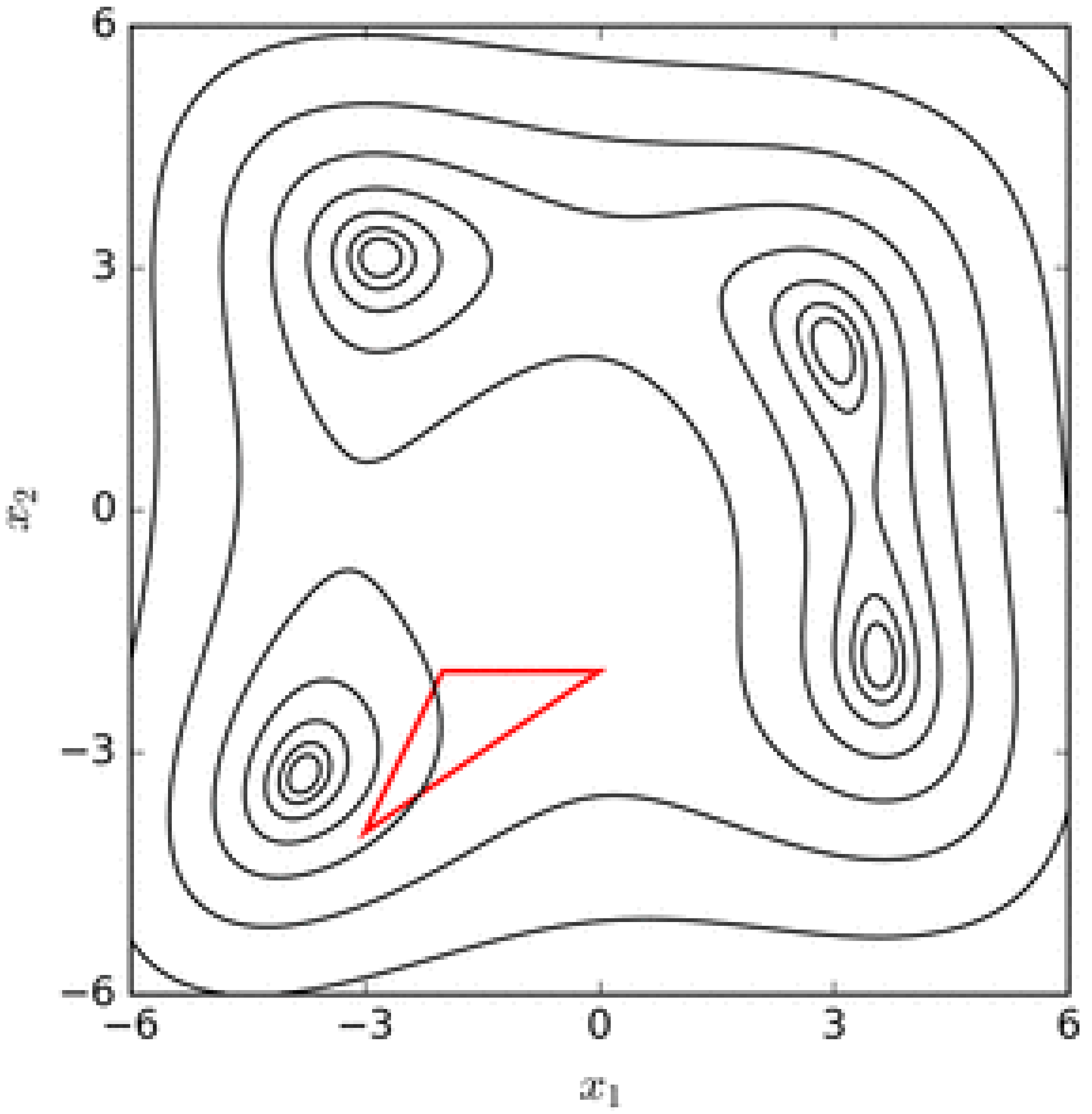
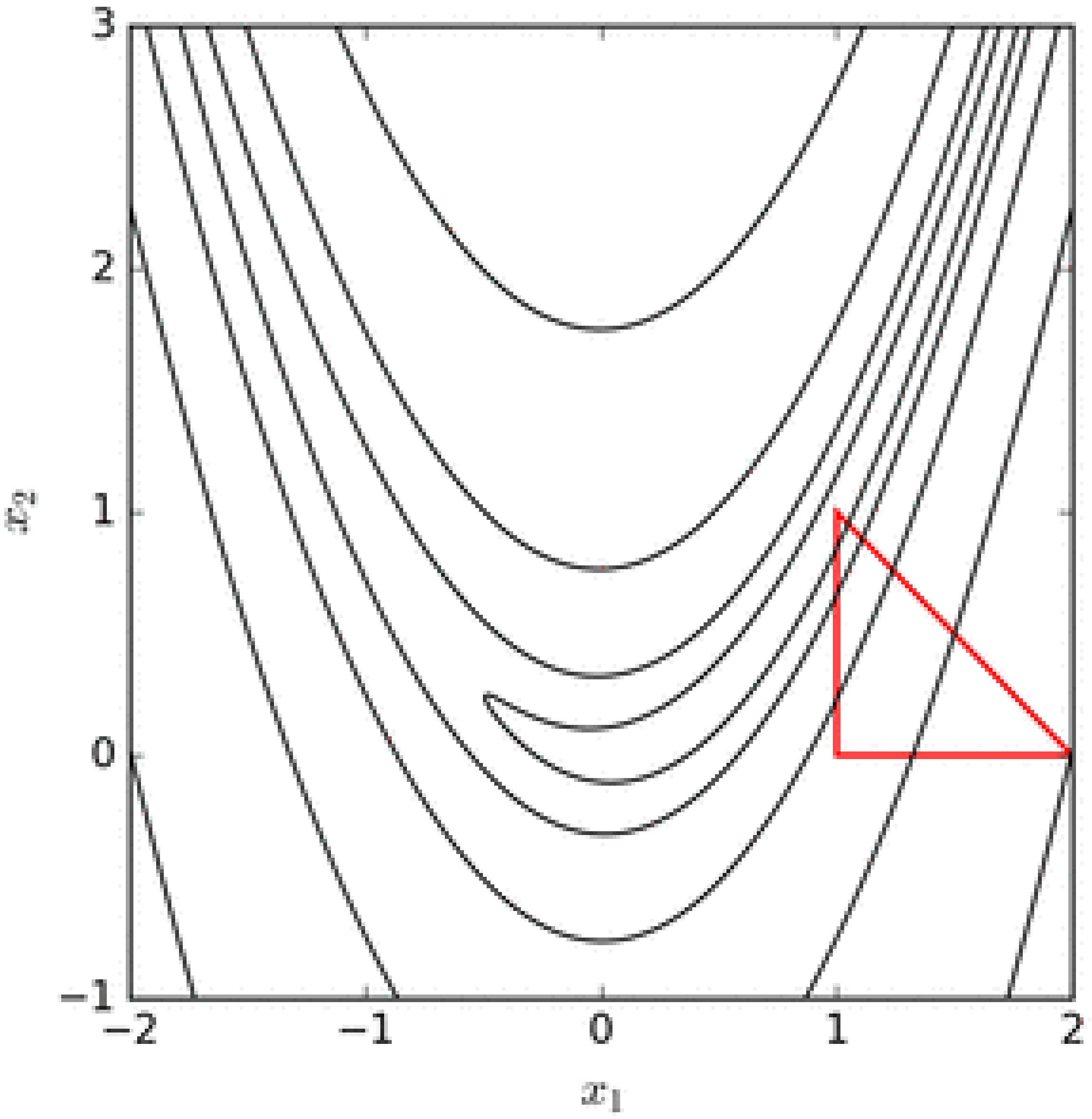


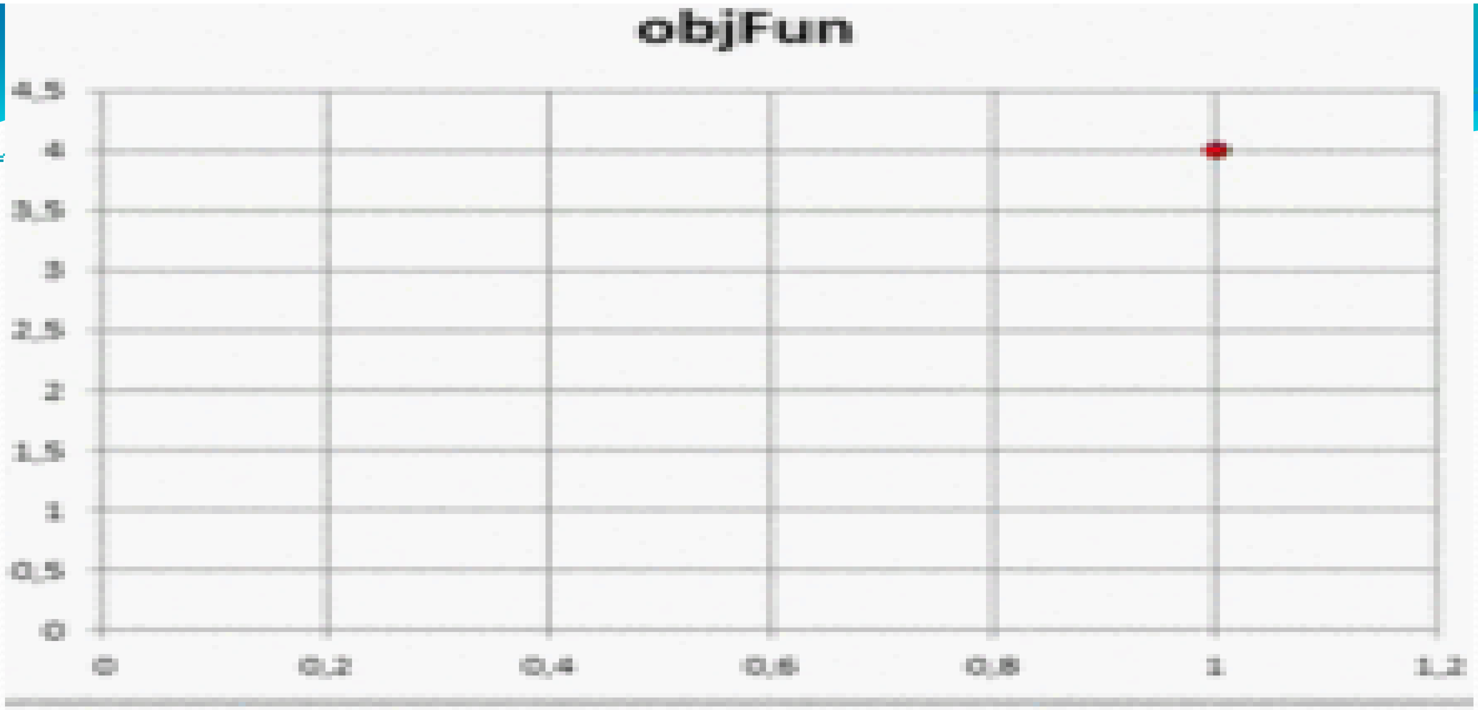
Figure 8.10 The sequence of triangles $\{T_k\}$ converging to the point $(3, 2)$ for the Nelder-Mead method.

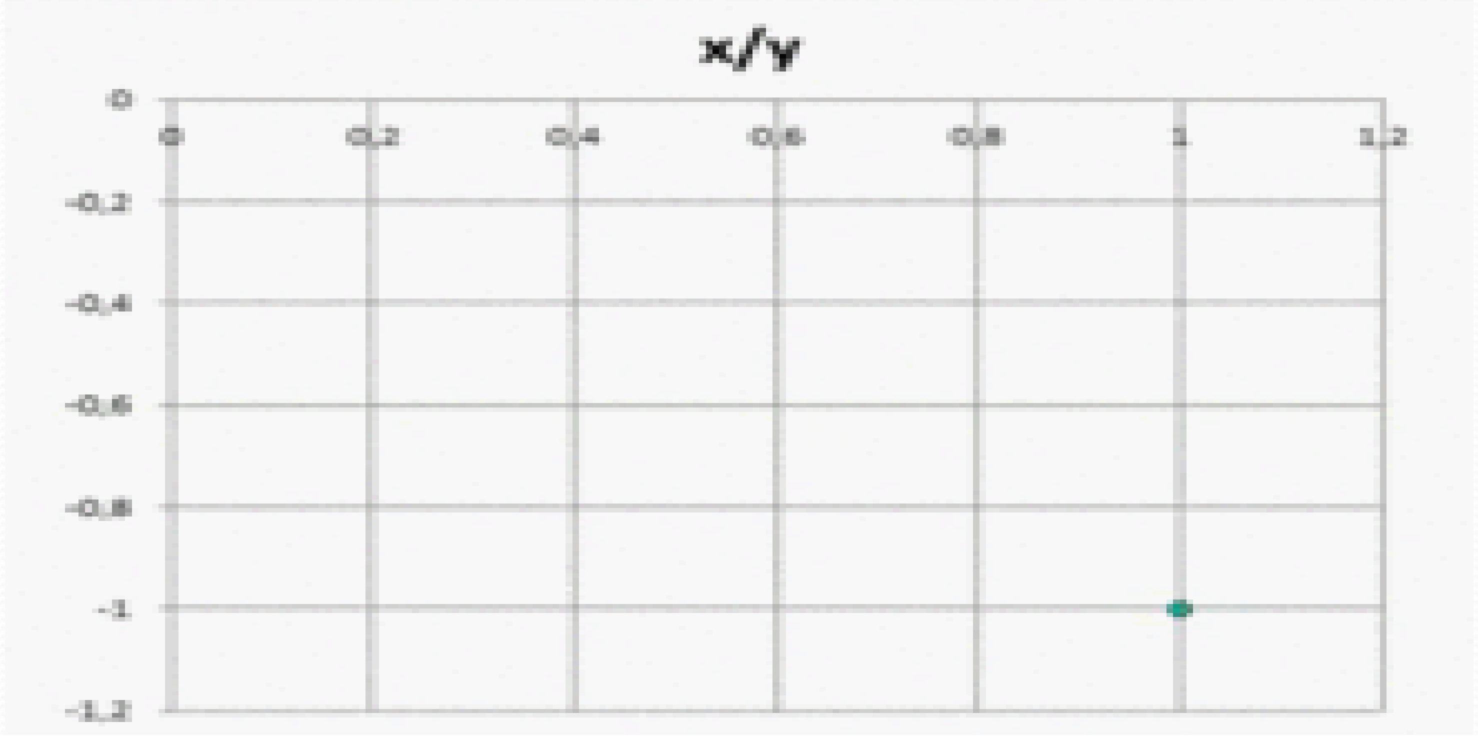


6
3
0
−3
−6
x_2
−6
−3
0
3
6
x_1



3
2
1
0
−1
−2
−1
0
1
2
x_1
x_2


objFun
4,5
4
3,5
3
2,5
2
1,5
1
0,5
0
0
0,2
0,4
0,6
0,8
1
1,2


x/y
0
-0,2
-0,4
-0,6
-0,8
-1
-1,2
0
0,2
0,4
0,6
0,8
1
1,2

Indirect Search (Descent) Methods
Μέθοδοι καθόδου ή μέθοδοι κλίσης

Κλίση μιας συνάρτησης

The gradient of a function is an n-component vector given by

$$\underset{n\times 1}{\nabla f} = \begin{Bmatrix} \partial f/\partial x_1 \\ \partial f/\partial x_2 \\ \vdots \\ \partial f/\partial x_n \end{Bmatrix} \tag{6.56}$$

The gradient has a very important property. If we move along the gradient direction from any point in n-dimensional space, the function value increases at the fastest rate. Hence the gradient direction is called the *direction of steepest ascent*. Unfortunately, the direction of steepest ascent is a local property and not a global one. This is illustrated in Fig. 6.14, where the gradient vectors ∇f evaluated at points 1, 2, 3, and 4 lie along the directions $11'$, $22'$, $33'$, and $44'$, respectively. Thus the function value increases at the fastest rate in the direction $11'$ at point 1, but not at point 2. Similarly, the function value increases at the fastest rate in direction $22'(33')$ at point 2 (3), but not at point 3 (4). In other words, the direction of steepest ascent generally varies from point to point, and if we make infinitely small moves along the direction of steepest ascent, the path will be a curved line like the curve 1–2–3–4 in Fig. 6.14.

Theorem 6.3 The gradient vector represents the direction of steepest ascent.

Theorem 6.4 The maximum rate of change of f at any point $\mathbf{X}$ is equal to the magnitude of the gradient vector at the same point.

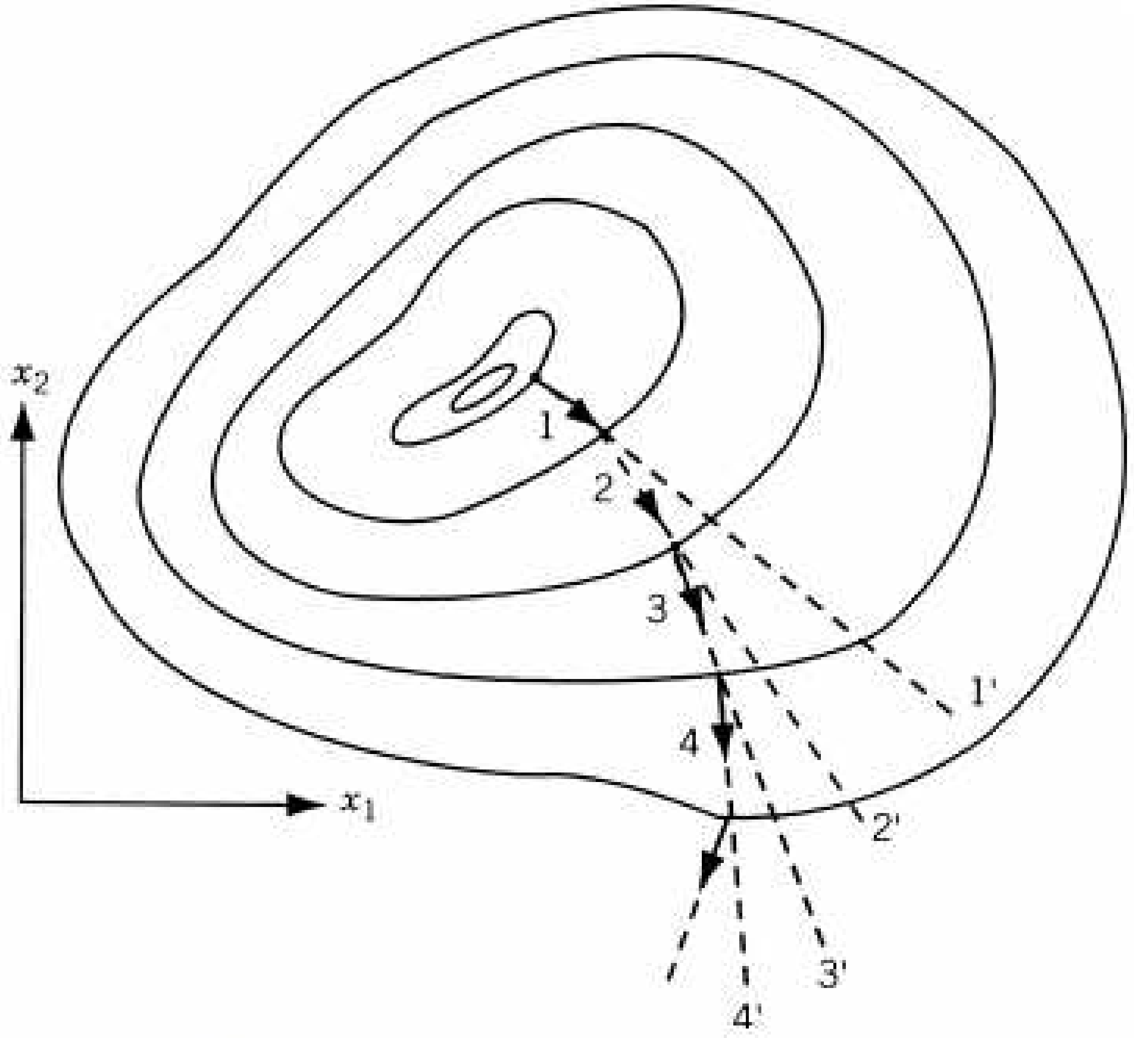


Figure 6.14 Steepest ascent directions.

The evaluation of the gradient requires the computation of the partial derivatives $\partial f/\partial x_i$, $i = 1, 2, \ldots, n$. There are three situations where the evaluation of the gradient poses certain problems:

1. The function is differentiable at all the points, but the calculation of the components of the gradient, $\partial f/\partial x_i$, is either impractical or impossible.
2. The expressions for the partial derivatives $\partial f/\partial x_i$ can be derived, but they require large computational time for evaluation.
3. The gradient ∇f is not defined at all the points.

In the first case, we can use the forward finite-difference formula

$$\left.\frac{\partial f}{\partial x_i}\right|_{\mathbf{X}_m} \simeq \frac{f(\mathbf{X}_m + \Delta x_i \mathbf{u}_i) - f(\mathbf{X}_m)}{\Delta x_i}, \quad i = 1, 2, \ldots, n \tag{6.63}$$

to approximate the partial derivative $\partial f/\partial x_i$ at $\mathbf{X}_m$. If the function value at the base point $\mathbf{X}_m$ is known, this formula requires one additional function evaluation to find $(\partial f/\partial x_i)|_{Xm}$. Thus it requires n additional function evaluations to evaluate the approximate gradient $\nabla f|_{\mathbf{X}m}$. For better results we can use the central finite difference formula to find the approximate partial derivative $\partial f/\partial x_i|_{\mathbf{X}m}$:

$$\left.\frac{\partial f}{\partial x_i}\right|_{\mathbf{X}_m} \simeq \frac{f(\mathbf{X}_m + \Delta x_i \mathbf{u}_i) - f(\mathbf{X}_m - \Delta x_i \mathbf{u}_i)}{2\Delta x_i}, \quad i = 1, 2, \ldots, n \tag{6.64}$$

2. The expressions for the partial derivatives $\partial f/\partial x_i$ can be derived, but they require large computational time for evaluation.
3. The gradient ∇f is not defined at all the points.

In the second case also, the use of finite-difference formulas is preferred whenever the exact gradient evaluation requires more computational time than the one involved in using Eq. (6.63) or (6.64).

In the third case, we cannot use the finite-difference formulas since the gradient is not defined at all the points. For example, consider the function shown in Fig. 6.15. If Eq. (6.64) is used to evaluate the derivative df/ds at $\mathbf{X}_m$, we obtain a value of α_1 for a step size Δx_1 and a value of α_2 for a step size Δx_2. Since, in reality, the derivative does not exist at the point $\mathbf{X}_m$, use of finite-difference formulas might lead to a complete breakdown of the minimization process. In such cases the minimization can be done only by one of the direct search techniques discussed earlier.

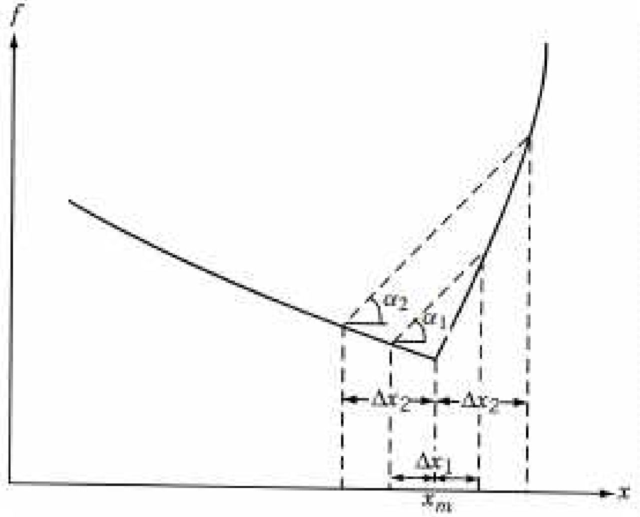


Figure 6.15 Gradient not defined at x_m.

STEEPEST DESCENT (CAUCHY) METHOD

The use of the negative of the gradient vector as a direction for minimization was first made by Cauchy in 1847 [6.12]. In this method we start from an initial trial point $\mathbf{X}_1$ and iteratively move along the steepest descent directions until the optimum point is found. The steepest descent method can be summarized by the following steps:

1. Start with an arbitrary initial point $\mathbf{X}_1$. Set the iteration number as $i = 1$.
2. Find the search direction $\mathbf{S}_i$ as

$$\mathbf{S}_i = -\nabla f_i = -\nabla f(\mathbf{X}_i) \tag{6.69}$$

3. Determine the optimal step length λ_i^* in the direction $\mathbf{S}_i$ and set

$$\mathbf{X}_{i+1} = \mathbf{X}_i + \lambda_i^* \mathbf{S}_i = \mathbf{X}_i - \lambda_i^* \nabla f_i \tag{6.70}$$

4. Test the new point, $\mathbf{X}_{i+1}$, for optimality. If $\mathbf{X}_{i+1}$ is optimum, stop the process. Otherwise, go to step 5.
5. Set the new iteration number $i = i + 1$ and go to step 2.

The method of steepest descent may appear to be the *best unconstrained minimization* technique since each one-dimensional search starts in the "best" direction. However, owing to the fact that the steepest descent direction is a local property, the method is not really effective in most problems.

Example 6.8 Minimize $f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$ starting from the point $\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$.

SOLUTION

Iteration 1

The gradient of f is given by

$$\nabla f = \begin{Bmatrix} \partial f/\partial x_1 \\ \partial f/\partial x_2 \end{Bmatrix} = \begin{Bmatrix} 1 + 4x_1 + 2x_2 \\ -1 + 2x_1 + 2x_2 \end{Bmatrix}$$

$$\nabla f_1 = \nabla f(\mathbf{X}_1) = \begin{Bmatrix} 1 \\ -1 \end{Bmatrix}$$

Therefore,

$$\mathbf{S}_1 = -\nabla f_1 = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix}$$

To find $\mathbf{X}_2$, we need to find the optimal step length λ_1^*. For this, we minimize $f(\mathbf{X}_1 + \lambda_1 \mathbf{S}_1) = f(-\lambda_1, \lambda_1) = \lambda_1^2 - 2\lambda_1$ with respect to λ_1. Since $df/d\lambda_1 = 0$ at $\lambda_1^* = 1$, we obtain

$$\mathbf{X}_2 = \mathbf{X}_1 + \lambda_1^* \mathbf{S}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} + 1 \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix}$$

As $\nabla f_2 = \nabla f(\mathbf{X}_2) = \begin{Bmatrix} -1 \\ -1 \end{Bmatrix} \neq \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$, $\mathbf{X}_2$ is not optimum.

Iteration 2

$$\mathbf{S}_2 = -\nabla f_2 = \begin{Bmatrix} 1 \\ 1 \end{Bmatrix}$$

To minimize

$$\begin{aligned} f(\mathbf{X}_2 + \lambda_2 \mathbf{S}_2) &= f(-1 + \lambda_2, 1 + \lambda_2) \\ &= 5\lambda_2^2 - 2\lambda_2 - 1 \end{aligned}$$

we set $df/d\lambda_2 = 0$. This gives $\lambda_2^* = \frac{1}{5}$, and hence

$$\mathbf{X}_3 = \mathbf{X}_2 + \lambda_2^* \mathbf{S}_2 = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} + \frac{1}{5} \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} = \begin{Bmatrix} -0.8 \\ 1.2 \end{Bmatrix}$$

Since the components of the gradient at $\mathbf{X}_3$, $\nabla f_3 = \begin{Bmatrix} 0.2 \\ -0.2 \end{Bmatrix}$, are not zero, we proceed to the next iteration.

Iteration 3

$$\mathbf{S}_3 = -\nabla f_3 = \begin{Bmatrix} -0.2 \\ 0.2 \end{Bmatrix}$$

As

$$f(\mathbf{X}_3 + \lambda_3 \mathbf{S}_3) = f(-0.8 - 0.2\lambda_3, 1.2 + 0.2\lambda_3)$$
$$= 0.04\lambda_3^2 - 0.08\lambda_3 - 1.20, \quad \frac{df}{d\lambda_3} = 0 \text{ at } \lambda_3^* = 1.0$$

Therefore,

$$\mathbf{X}_4 = \mathbf{X}_3 + \lambda_3^* \mathbf{S}_3 = \begin{Bmatrix} -0.8 \\ 1.2 \end{Bmatrix} + 1.0 \begin{Bmatrix} -0.2 \\ 0.2 \end{Bmatrix} = \begin{Bmatrix} -1.0 \\ 1.4 \end{Bmatrix}$$

The gradient at $\mathbf{X}_4$ is given by

$$\nabla f_4 = \begin{Bmatrix} -0.20 \\ -0.20 \end{Bmatrix}$$

Since $\nabla f_4 \neq \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$, $\mathbf{X}_4$ is not optimum and hence we have to proceed to the next iteration. This process has to be continued until the optimum point, $\mathbf{X}^* = \begin{Bmatrix} -1.0 \\ 1.5 \end{Bmatrix}$, is found.

Convergence Criteria: The following criteria can be used to terminate the iterative process.

1. When the change in function value in two consecutive iterations is small:

$$\left|\frac{f(\mathbf{X}_{i+1}) - f(\mathbf{X}_i)}{f(\mathbf{X}_i)}\right| \le \varepsilon_1 \tag{6.71}$$

2. When the partial derivatives (components of the gradient) of f are small:

$$\left|\frac{\partial f}{\partial x_i}\right| \le \varepsilon_2, \quad i = 1, 2, \ldots, n \tag{6.72}$$

3. When the change in the design vector in two consecutive iterations is small:

$$|\mathbf{X}_{i+1} - \mathbf{X}_i| \le \varepsilon_3 \tag{6.73}$$

CONJUGATE GRADIENT (FLETCHER–REEVES) METHOD

The iterative procedure of Fletcher–Reeves method can be stated as follows:

1. Start with an arbitrary initial point $\mathbf{X}_1$.
2. Set the first search direction $\mathbf{S}_1 = -\nabla f(\mathbf{X}_1) = -\nabla f_1$.
3. Find the point $\mathbf{X}_2$ according to the relation

$$\mathbf{X}_2 = \mathbf{X}_1 + \lambda_1^* \mathbf{S}_1 \tag{6.80}$$

   where λ_1^* is the optimal step length in the direction $\mathbf{S}_1$. Set $i = 2$ and go to the next step.

4. Find $\nabla f_i = \nabla f(\mathbf{X}_i)$, and set

$$\mathbf{S}_i = -\nabla f_i + \frac{|\nabla f_i|^2}{|\nabla f_{i-1}|^2}\mathbf{S}_{i-1} \tag{6.81}$$

5. Compute the optimum step length λ_i^* in the direction $\mathbf{S}_i$, and find the new point

$$\mathbf{X}_{i+1} = \mathbf{X}_i + \lambda_i^* \mathbf{S}_i \tag{6.82}$$

6. Test for the optimality of the point $\mathbf{X}_{i+1}$. If $\mathbf{X}_{i+1}$ is optimum, stop the process. Otherwise, set the value of $i = i + 1$ and go to step 4.

Remarks:

1. The Fletcher–Reeves method was originally proposed by Hestenes and Stiefel [6.14] as a method for solving systems of linear equations derived from the stationary conditions of a quadratic. Since the directions $\mathbf{S}_i$ used in this method are $\mathbf{A}$-conjugate, the process should converge in n cycles or less for a quadratic function. However, for ill-conditioned quadratics (whose contours are highly eccentric and distorted), the method may require much more than n cycles for convergence. The reason for this has been found to be the cumulative effect of rounding errors. Since $\mathbf{S}_i$ is given by Eq. (6.81), any error resulting from the inaccuracies involved in the determination of λ_i^*, and from the round-off error involved in accumulating the successive $|\nabla f_i|^2 \mathbf{S}_{i-1}/|\nabla f_{i-1}|^2$ terms, is carried forward through the vector $\mathbf{S}_i$. Thus the search directions $\mathbf{S}_i$ will be progressively contaminated by these errors. Hence it is necessary, in practice, to restart the method periodically after every, say, m steps by taking the new search direction as the steepest descent direction. That is, after every m steps, $\mathbf{S}_{m+1}$ is set equal to $-\nabla f_{m+1}$ instead of the usual form. Fletcher and Reeves have recommended a value of $m = n + 1$, where n is the number of design variables.
2. Despite the limitations indicated above, the Fletcher–Reeves method is vastly superior to the steepest descent method and the pattern search methods, but it turns out to be rather less efficient than the Newton and the quasi-Newton (variable metric) methods discussed in the latter sections.

Example 6.9 Minimize $f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$ starting from the point $\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$.

SOLUTION

Iteration 1

$$\nabla f = \begin{Bmatrix} \partial f/\partial x_1 \\ \partial f/\partial x_2 \end{Bmatrix} = \begin{Bmatrix} 1 + 4x_1 + 2x_2 \\ -1 + 2x_1 + 2x_2 \end{Bmatrix}$$

$$\nabla f_1 = \nabla f(\mathbf{X}_1) = \begin{Bmatrix} 1 \\ -1 \end{Bmatrix}$$

The search direction is taken as $\mathbf{S}_1 = -\nabla f_1 = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix}$. To find the optimal step length λ_1^* along $\mathbf{S}_1$, we minimize $f(\mathbf{X}_1 + \lambda_1 \mathbf{S}_1)$ with respect to λ_1. Here

$$f(\mathbf{X}_1 + \lambda_1 \mathbf{S}_1) = f(-\lambda_1, +\lambda_1) = \lambda_1^2 - 2\lambda_1$$

$$\frac{df}{d\lambda_1} = 0 \quad \text{at} \quad \lambda_1^* = 1$$

Therefore,

$$\mathbf{X}_2 = \mathbf{X}_1 + \lambda_1^* \mathbf{S}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} + 1 \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix}$$

Iteration 2

Since $\nabla f_2 = \nabla f(\mathbf{X}_2) = \begin{Bmatrix} -1 \\ -1 \end{Bmatrix}$, Eq. (6.81) gives the next search direction as

$$\mathbf{S}_2 = -\nabla f_2 + \frac{|\nabla f_2|^2}{|\nabla f_1|^2}\mathbf{S}_1$$

where

$$|\nabla f_1|^2 = 2 \quad \text{and} \quad |\nabla f_2|^2 = 2$$

Therefore,

$$\mathbf{S}_2 = -\begin{Bmatrix} -1 \\ -1 \end{Bmatrix} + \left(\frac{2}{2}\right)\begin{Bmatrix} -1 \\ 1 \end{Bmatrix} = \begin{Bmatrix} 0 \\ +2 \end{Bmatrix}$$

To find λ_2^*, we minimize

$$\begin{aligned} f(\mathbf{X}_2 + \lambda_2\mathbf{S}_2) &= f(-1, 1 + 2\lambda_2) \\ &= -1 - (1 + 2\lambda_2) + 2 - 2(1 + 2\lambda_2) + (1 + 2\lambda_2)^2 \\ &= 4\lambda_2^2 - 2\lambda_2 - 1 \end{aligned}$$

with respect to λ_2. As $df/d\lambda_2 = 8\lambda_2 - 2 = 0$ at $\lambda_2^* = \frac{1}{4}$, we obtain

$$\mathbf{X}_3 = \mathbf{X}_2 + \lambda_2^*\mathbf{S}_2 = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} + \frac{1}{4}\begin{Bmatrix} 0 \\ 2 \end{Bmatrix} = \begin{Bmatrix} -1 \\ 1.5 \end{Bmatrix}$$

Thus the optimum point is reached in two iterations. Even if we do not know this point to be optimum, we will not be able to move from this point in the next iteration. This can be verified as follows.

Iteration 3

Now

$$\nabla f_3 = \nabla f(\mathbf{X}_3) = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}, \quad |\nabla f_2|^2 = 2, \quad \text{and} \quad |\nabla f_3|^2 = 0.$$

Thus

$$\mathbf{S}_3 = -\nabla f_3 + (|\nabla f_3|^2/|\nabla f_2|^2)\mathbf{S}_2 = -\begin{Bmatrix} 0 \\ 0 \end{Bmatrix} + \left(\frac{0}{2}\right)\begin{Bmatrix} 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

This shows that there is no search direction to reduce f further, and hence $\mathbf{X}_3$ is optimum.

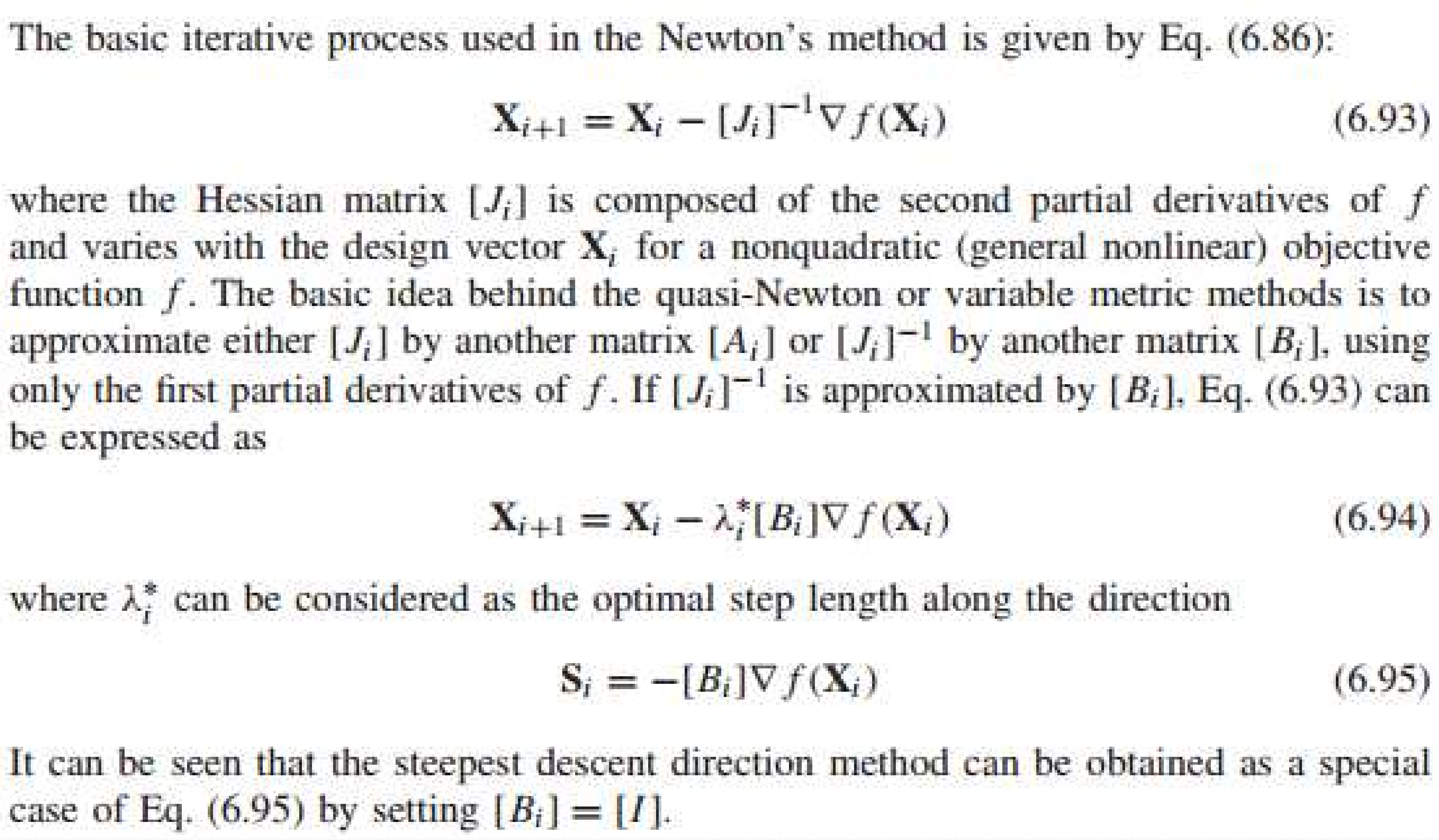
Μέθοδος Newton

- Στην προηγούμενη διάλεξη παρουσιάστηκε η μέθοδος Newton για τη βελτιστοποίηση μονοδιάστατων προβλημάτων.
- Η μεθοδολογία προσέγγισης της συνάρτησης με ανάπτυξη τετραγωνικής σειράς Taylor μπορεί να εφαρμοστεί και σε συναρτήσεις πολλών μεταβλητών.
- Έπειτα υπολογίζεται αναλυτικά το βέλτιστο της προσεγγιστικής συνάρτησης και μ' αυτόν τον τρόπο προσεγγίζεται πολύ πιο γρήγορα το πραγματικό βέλτιστο.
- Για τετραγωνικές συναρτήσεις το βέλτιστο εξάγεται σε 1 βήμα. Για μη τετραγωνικές συναρτήσεις η μέθοδος Newton μπορεί να αποκλίνει ή να συγκλίνει σε σημείο σάγματος (saddle point).

Μέθοδος Marquardt

- Η μέθοδος της μέγιστης καθόδου (steepest descent method) ελαττώνει την αντικειμενική συνάρτηση όταν το υφιστάμενο σημείο $\mathbf{X_i}$ βρίσκεται μακριά από το βέλτιστο $\mathbf{X}^*$. Η μέθοδος Newton συγκλίνει γρήγορα όταν βρίσκεται κοντά στο βέλτιστο.
- Η μέθοδος Marquardt επιδιώκει να αποκτήσει τα πλεονεκτήματα των δύο παραπάνω μεθόδων. Τροποποιώντας τα διαγώνια στοιχεία του Εσσιανού πίνακα (Hessian matrix) προστίθεται ένας θετικός σταθερός αριθμός $α_i$, ο οποίος όταν έχει μεγάλες τιμές η αναζήτηση λειτουργεί παρόμοια με τη steepest descent method. Όταν ο $α_i$ έχει μικρές τιμές τότε η αναζήτηση λειτουργεί παρόμοια με τη μέθοδο Newton.
- Επομένως μια καλή λειτουργία είναι η εκκίνηση της αναζήτησης με μεγάλες τιμές $α_i$ και καθώς θα εξελίσσεται να μειώνονται οι τιμές του $α_i$ προσεγγίζοντας το μηδέν.

QUASI-NEWTON METHODS


The basic iterative process used in the Newton's method is given by Eq. (6.86):

$$\mathbf{X}_{i+1} = \mathbf{X}_i - [J_i]^{-1}\nabla f(\mathbf{X}_i) \tag{6.93}$$

where the Hessian matrix $[J_i]$ is composed of the second partial derivatives of f and varies with the design vector $\mathbf{X}_i$ for a nonquadratic (general nonlinear) objective function f. The basic idea behind the quasi-Newton or variable metric methods is to approximate either $[J_i]$ by another matrix $[A_i]$ or $[J_i]^{-1}$ by another matrix $[B_i]$, using only the first partial derivatives of f. If $[J_i]^{-1}$ is approximated by $[B_i]$, Eq. (6.93) can be expressed as

$$\mathbf{X}_{i+1} = \mathbf{X}_i - \lambda_i^*[B_i]\nabla f(\mathbf{X}_i) \tag{6.94}$$

where λ_i^* can be considered as the optimal step length along the direction

$$\mathbf{S}_i = -[B_i]\nabla f(\mathbf{X}_i) \tag{6.95}$$

It can be seen that the steepest descent direction method can be obtained as a special case of Eq. (6.95) by setting $[B_i] = [I]$.

Υπάρχουν διάφοροι μέθοδοι υπολογισμού για την ανανέωση του πίνακα B_i

DAVIDON–FLETCHER–POWELL METHOD

The iterative procedure of the Davidon–Fletcher–Powell (DFP) method can be described as follows:

1. Start with an initial point $\mathbf{X}_1$ and a $n \times n$ positive definite symmetric matrix $[B_1]$ to approximate the inverse of the Hessian matrix of f. Usually, $[B_1]$ is taken as the identity matrix $[I]$. Set the iteration number as $i = 1$.
2. Compute the gradient of the function, ∇f_i, at point $\mathbf{X}_i$, and set

$$\mathbf{S}_i = -[B_i]\nabla f_i \tag{6.128}$$

3. Find the optimal step length λ_i^* in the direction $\mathbf{S}_i$ and set

$$\mathbf{X}_{i+1} = \mathbf{X}_i + \lambda_i^* \mathbf{S}_i \tag{6.129}$$

4. Test the new point $\mathbf{X}_{i+1}$ for optimality. If $\mathbf{X}_{i+1}$ is optimal, terminate the iterative process. Otherwise, go to step 5.
5. Update the matrix $[B_i]$ using Eq. (6.119) as

$$[B_{i+1}] = [B_i] + [M_i] + [N_i] \tag{6.130}$$

where

$$[M_i] = \lambda_i^* \frac{\mathbf{S}_i \mathbf{S}_i^{\mathrm{T}}}{\mathbf{S}_i^T \mathbf{g}_i} \tag{6.131}$$

$$[N_i] = -\frac{([B_i]\mathbf{g}_i)([B_i]\mathbf{g}_i)^{\mathrm{T}}}{\mathbf{g}_i^T [B_i]\mathbf{g}_i} \tag{6.132}$$

$$\mathbf{g}_i = \nabla f(\mathbf{X}_{i+1}) - \nabla f(\mathbf{X}_i) = \nabla f_{i+1} - \nabla f_i \tag{6.133}$$

6. Set the new iteration number as $i = i + 1$, and go to step 2.

BROYDEN–FLETCHER–GOLDFARB–SHANNO METHOD

As stated earlier, a major difference between the DFP and BFGS methods is that in the BFGS method, the Hessian matrix is updated iteratively rather than the inverse of the Hessian matrix. The BFGS method can be described by the following steps.

1. Start with an initial point $\mathbf{X}_1$ and a $n \times n$ positive definite symmetric matrix $[B_1]$ as an initial estimate of the inverse of the Hessian matrix of f. In the absence of additional information, $[B_1]$ is taken as the identity matrix $[I]$. Compute the gradient vector $\nabla f_1 = \nabla f(\mathbf{X}_1)$ and set the iteration number as $i = 1$.
2. Compute the gradient of the function, ∇f_i, at point $\mathbf{X}_i$, and set

$$\mathbf{S}_i = -[B_i]\nabla f_i \tag{6.134}$$

3. Find the optimal step length λ_i^* in the direction $\mathbf{S}_i$ and set

$$\mathbf{X}_{i+1} = \mathbf{X}_i + \lambda_i^* \mathbf{S}_i \tag{6.135}$$

4. Test the point $\mathbf{X}_{i+1}$ for optimality. If $\|\nabla f_{i+1}\| \le \varepsilon$, where ε is a small quantity, take $\mathbf{X}^* \approx \mathbf{X}_{i+1}$ and stop the process. Otherwise, go to step 5.
5. Update the Hessian matrix as

$$[B_{i+1}] = [B_i] + \left(1 + \frac{\mathbf{g}_i^T [B_i]\mathbf{g}_i}{\mathbf{d}_i^T \mathbf{g}_i}\right)\frac{\mathbf{d}_i \mathbf{d}_i^T}{\mathbf{d}_i^T \mathbf{g}_i} - \frac{\mathbf{d}_i \mathbf{g}_i^T [B_i]}{\mathbf{d}_i^T \mathbf{g}_i} - \frac{[B_i]\mathbf{g}_i \mathbf{d}_i^T}{\mathbf{d}_i^T \mathbf{g}_i} \tag{6.136}$$

where

$$\mathbf{d}_i = \mathbf{X}_{i+1} - \mathbf{X}_i = \lambda_i^* \mathbf{S}_i \tag{6.137}$$

$$\mathbf{g}_i = \nabla f_{i+1} - \nabla f_i = \nabla f(\mathbf{X}_{i+1}) - \nabla f(\mathbf{X}_i) \tag{6.138}$$

6. Set the new iteration number as $i = i + 1$ and go to step 2.

Remarks:

1. The BFGS method can be considered as a quasi-Newton, conjugate gradient, and variable metric method.
2. Since the inverse of the Hessian matrix is approximated, the BFGS method can be called an indirect update method.
3. If the step lengths λ_i^* are found accurately, the matrix, $[B_i]$, retains its positive definiteness as the value of i increases. However, in practical application, the matrix $[B_i]$ might become indefinite or even singular if λ_i^* are not found accurately. As such, periodical resetting of the matrix $[B_i]$ to the identity matrix $[I]$ is desirable. However, numerical experience indicates that the BFGS method is less influenced by errors in λ_i^* than is the DFP method.
4. It has been shown that the BFGS method exhibits superlinear convergence near $\mathbf{X}^*$ [6.19].

Δοκιμαστικές συναρτήσεις

- Η απόδοση των αλγορίθμων βελτιστοποίησης δοκιμάζεται σε τυπικές συναρτήσεις.
- Έχουν προταθεί διάφορες συναρτήσεις με διαφορετικά στοιχεία, όπως ο αριθμός των μεταβλητών, η πολυπλοκότητα, ο αριθμός των ακρότατων κ.α.
- Ο σκοπός των συναρτήσεων είναι να συγκριθεί η απόδοση των αλγορίθμων.
- Συνήθως η εκκίνηση των αλγορίθμων υλοποιείται από συγκεκριμένο σημείο.
- Ως κριτήριο σύγκρισης τυπικά είναι ο συνολικός αριθμός των εκτιμήσεων της αντικειμενικής συνάρτησης.

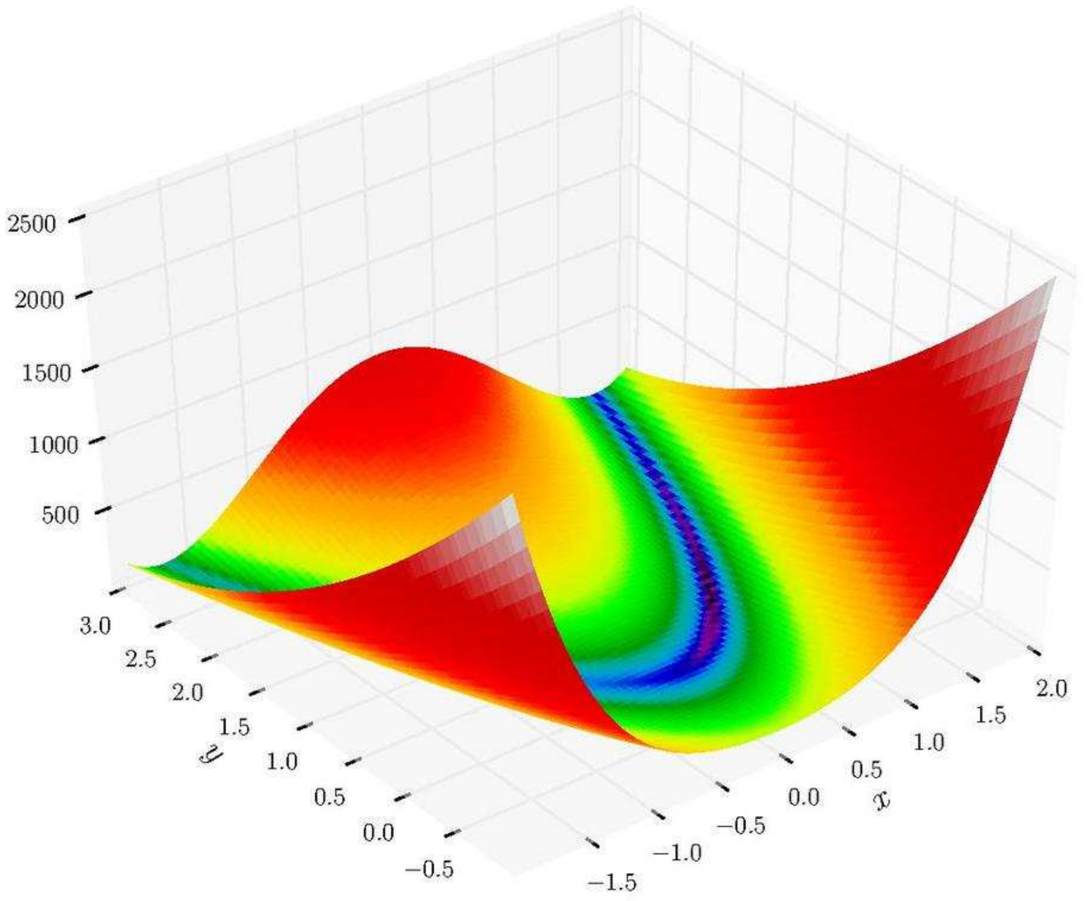
Δοκιμαστικές συναρτήσεις

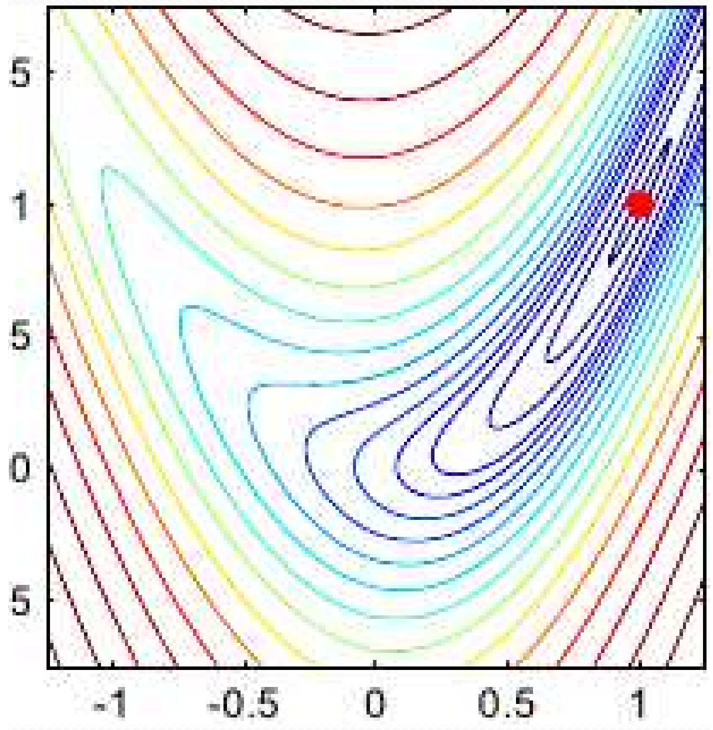
1. Rosenbrock's parabolic valley [6.8]:

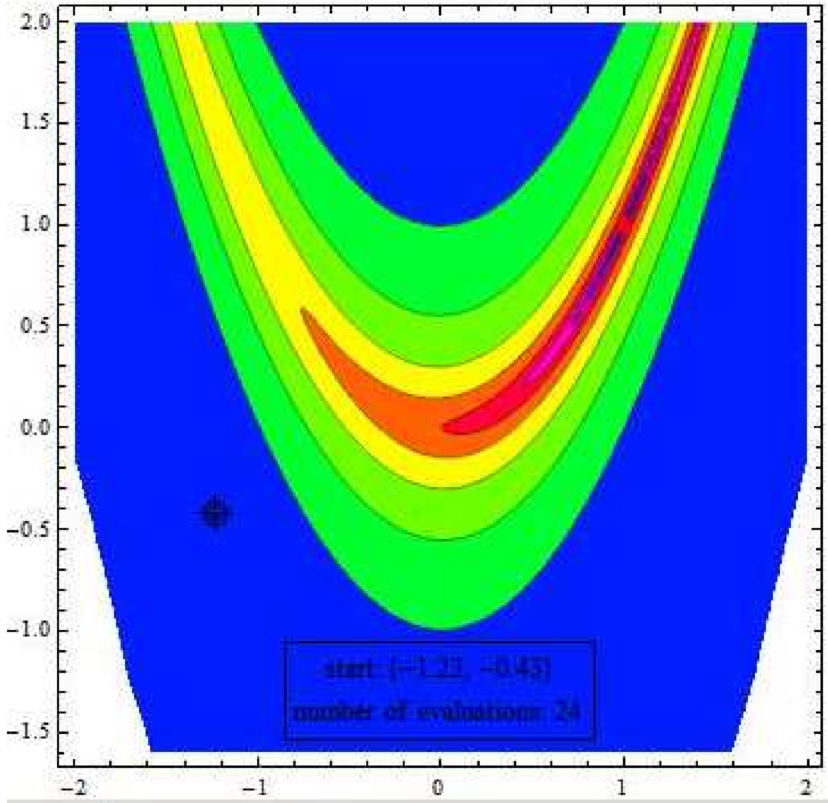
$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

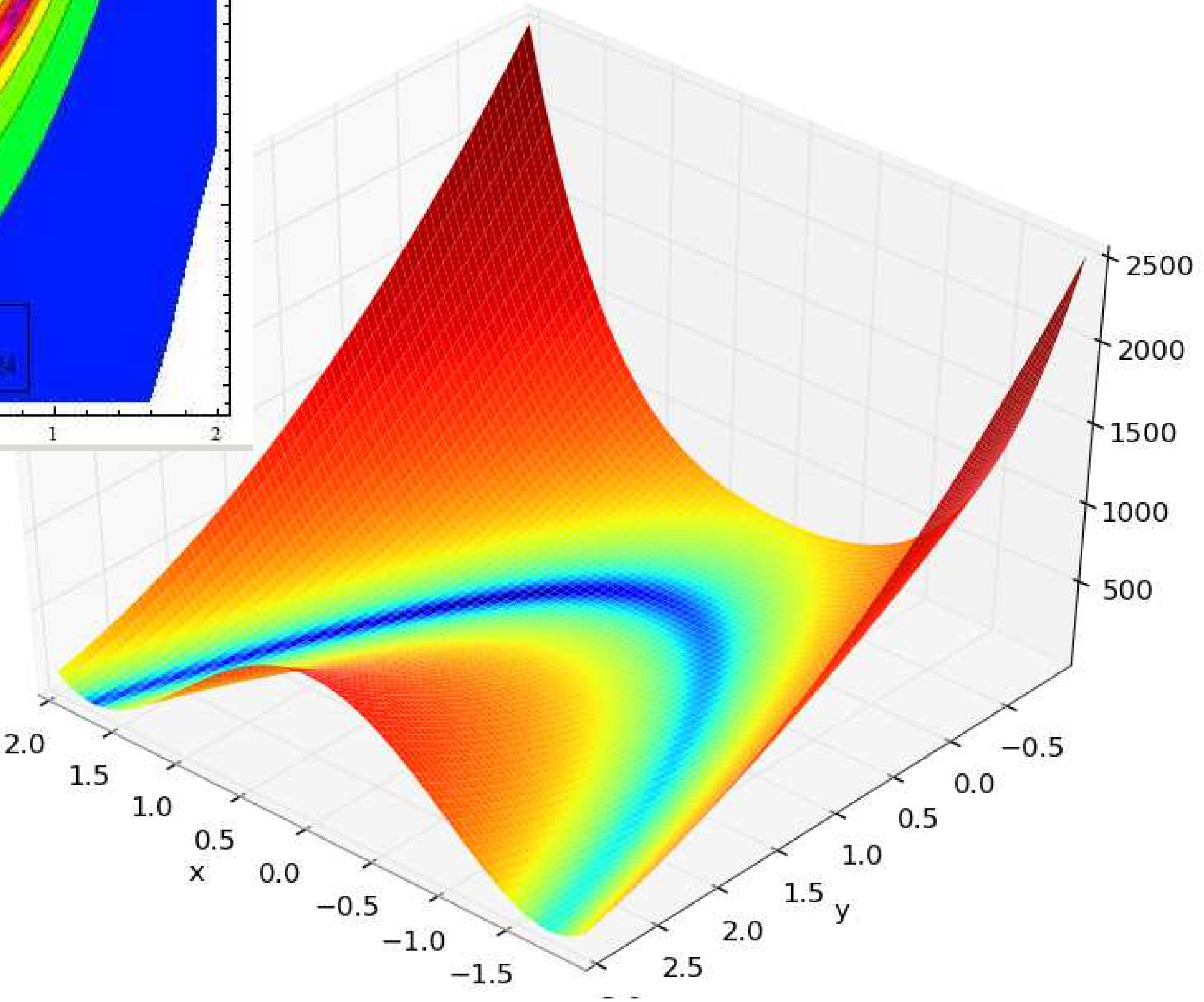
$$\mathbf{X}_1 = \begin{Bmatrix} -1.2 \\ 1.0 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 1 \\ 1 \end{Bmatrix}$$

$$f_1 = 24.0, \quad f^* = 0.0$$






start: [−1.23, −0.43]
number of evaluations: 24
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2

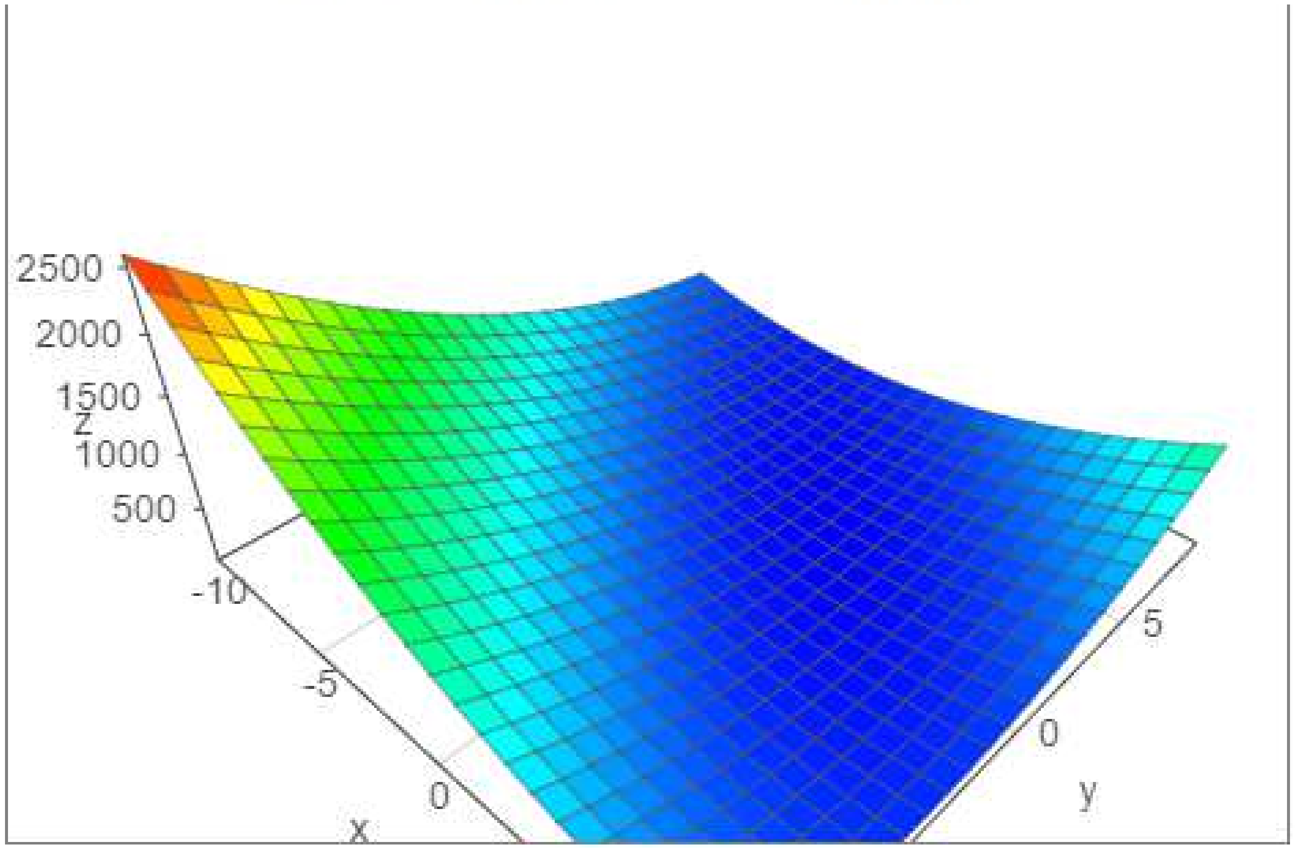

2500
2000
1500
1000
500
2.0
1.5
1.0
0.5
x
0.0
−0.5
−1.0
−1.5
−0.5
0.0
0.5
1.0
1.5
y
2.0
2.5

2. A quadratic function:

$$f(x_1, x_2) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$$

$$\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 1 \\ 3 \end{Bmatrix}$$

$$f_1 = 7.40, \quad f^* = 0.0$$



3. Powell's quartic function [6.7]:

$$f(x_1, x_2, x_3, x_4) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$$

$$\mathbf{X}_1^{\mathrm{T}} = \{x_1\ x_2\ x_3\ x_4\}_1 = \{3\ \ -1\ 0\ 1\}, \quad \mathbf{X}^{*\mathrm{T}} = \{0\ 0\ 0\ 0\}$$

$$f_1 = 215.0, \quad f^* = 0.0$$

4. Fletcher and Powell's helical valley [6.21]:

$$f(x_1, x_2, x_3) = 100 \left\{ [x_3 - 10\theta(x_1, x_2)]^2 + [\sqrt{x_1^2 + x_2^2} - 1]^2 \right\} + x$$

where

$$2\pi\theta(x_1, x_2) = \begin{cases} \arctan \dfrac{x_2}{x_1} & \text{if } x_1 > 0 \\ \pi + \arctan \dfrac{x_2}{x_1} & \text{if } x_1 < 0 \end{cases}$$

$$\mathbf{X}_1 = \begin{Bmatrix} -1 \\ 0 \\ 0 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix}$$

$$f_1 = 25{,}000.0, \quad f^* = 0.0$$

5. A nonlinear function of three variables [6.7]:

$$f(x_1, x_2, x_3) = \frac{1}{1+(x_1-x_2)^2} + \sin\left(\frac{1}{2}\pi x_2 x_3\right) + \exp\left[-\left(\frac{x_1+x_3}{x_2}-2\right)^2\right]$$

$$\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 1 \\ 2 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 1 \\ 1 \\ 1 \end{Bmatrix}$$

$$f_1 = 1.5, \quad f^* = f_{\max} = 3.0$$

10. Wood's function [6.30]:

$$f(x_1, x_2, x_3, x_4) = [10(x_2 - x_1^2)]^2 + (1-x_1)^2 + 90(x_4 - x_3^2)^2 + (1-x_3)^2 + 10(x_2 + x_4 - 2)^2 + 0.1(x_2 - x_4)$$

$$\mathbf{X}_1 = \begin{Bmatrix} -3 \\ -1 \\ -3 \\ -1 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{Bmatrix}$$

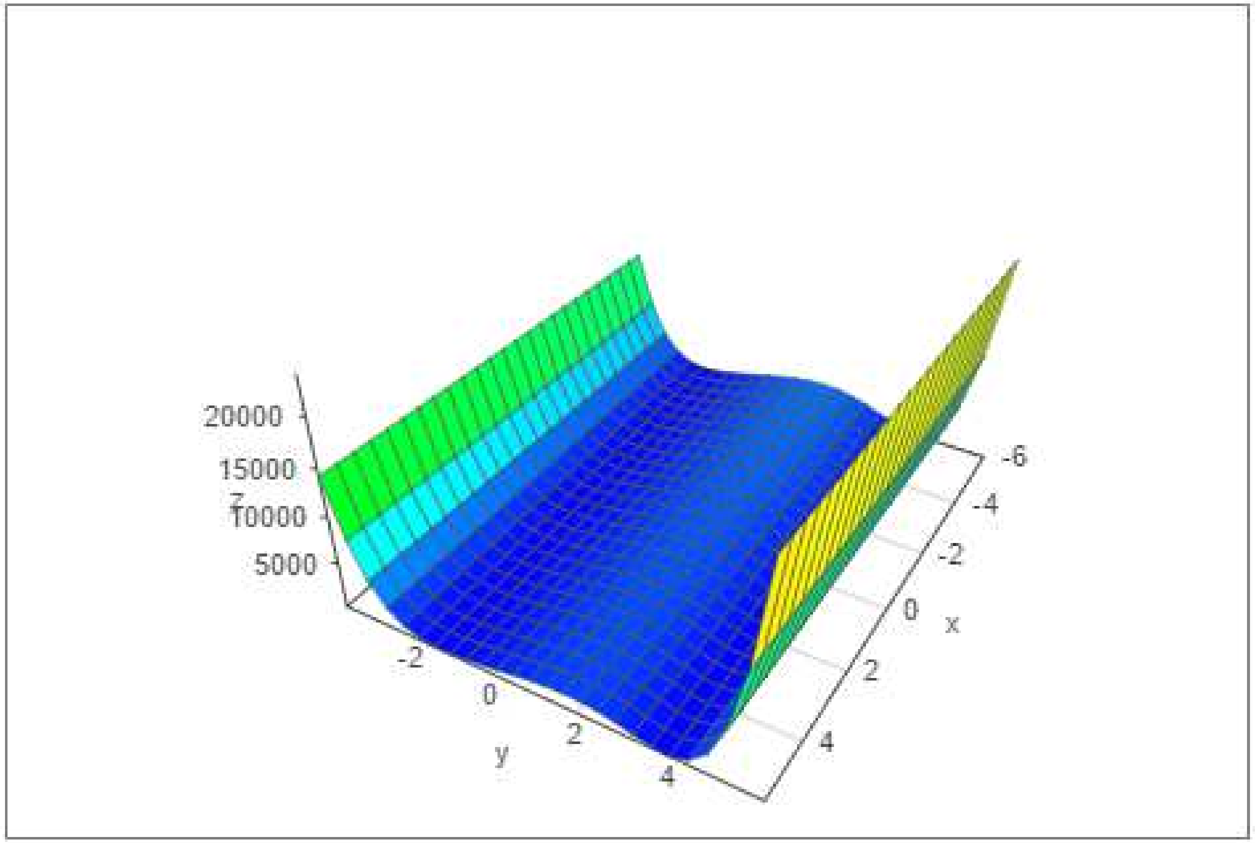
$$f_1 = 19192.0, \quad f^* = 0.0$$

6. Freudenstein and Roth function [6.27]:

$$f(x_1, x_2) = \{-13 + x_1 + [(5 - x_2)x_2 - 2]x_2\}^2 + \{-29 + x_1 + [(x_2 + 1)x_2 - 14]x_2\}^2$$

$$\mathbf{X}_1 = \begin{Bmatrix} 0.5 \\ -2 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 5 \\ 4 \end{Bmatrix}, \quad \mathbf{X}^*_{\text{alternate}} = \begin{Bmatrix} 11.41\ldots \\ -0.8968\ldots \end{Bmatrix}$$

$$f_1 = 400.5, \quad f^* = 0.0, \quad f^*_{\text{alternate}} = 48.9842\ldots$$

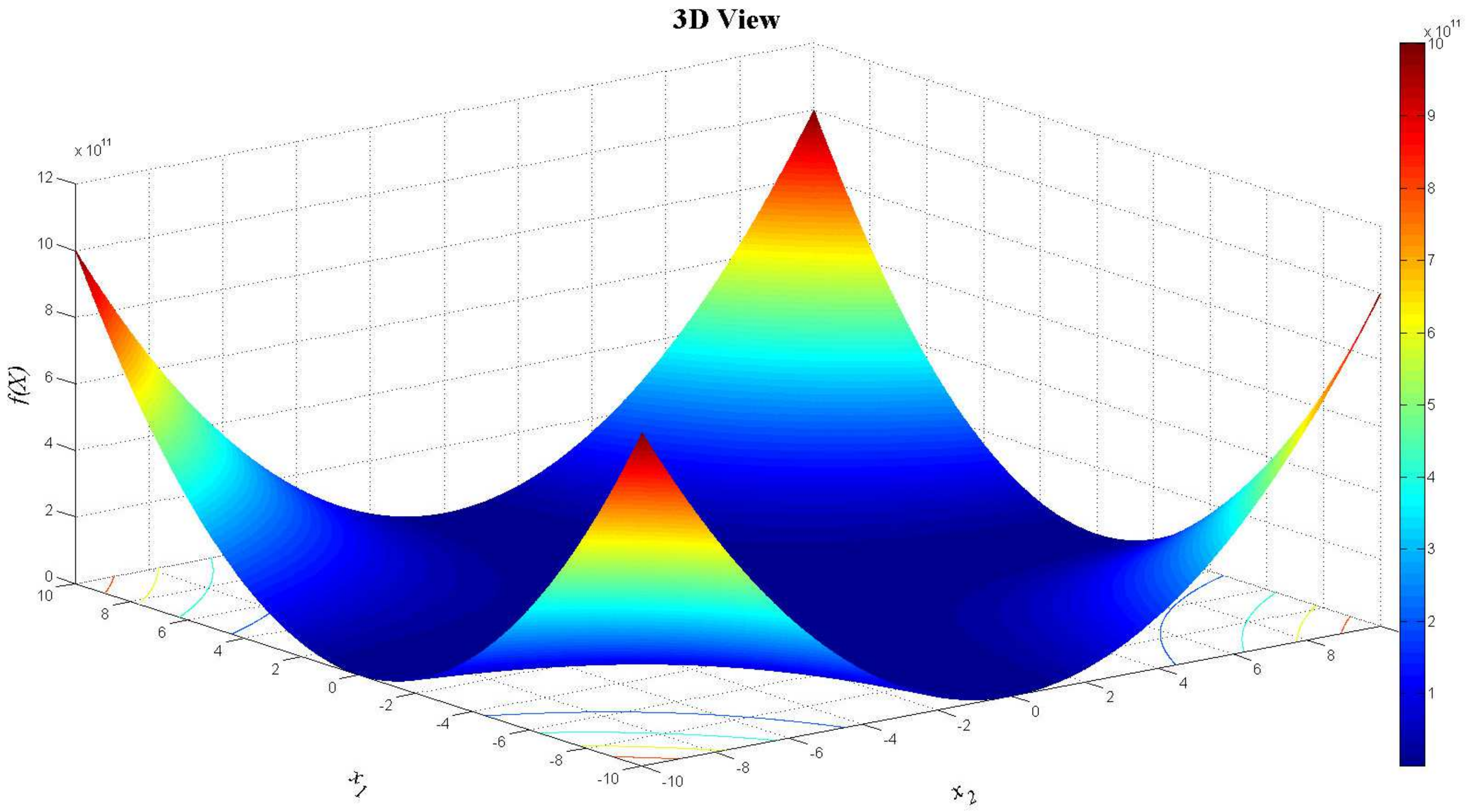


7. Powell's badly scaled function [6.28]:

$$f(x_1, x_2) = (10{,}000x_1x_2 - 1)^2 + [\exp(-x_1) + \exp(-x_2) - 1.0001]^2$$

$$\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 1.098\ldots \times 10^{-5} \\ 9.106\ldots \end{Bmatrix}$$

$$f_1 = 1.1354, \quad f^* = 0.0$$

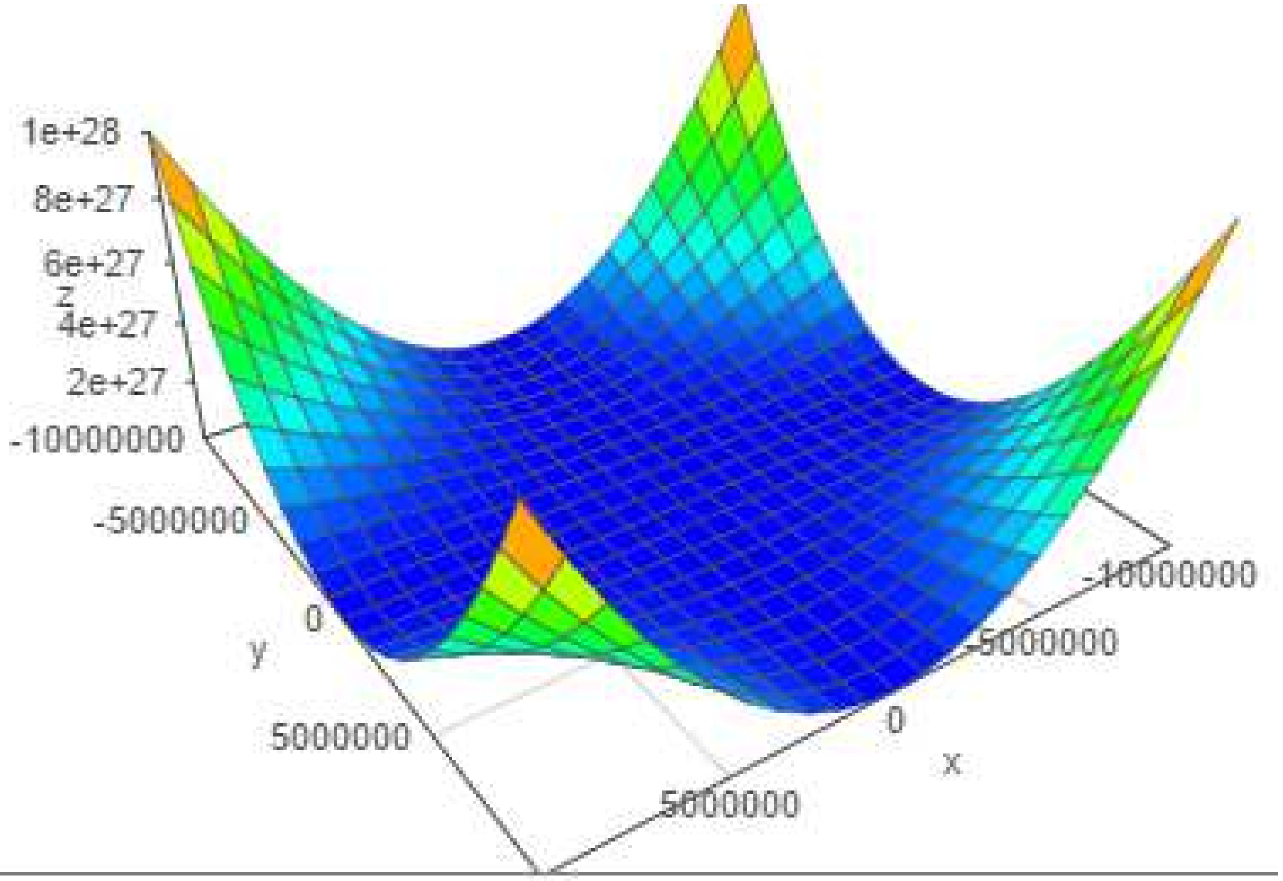


8. Brown's badly scaled function [6.29]:

$$f(x_1, x_2) = (x_1 - 10^6)^2 + (x_2 - 2 \times 10^{-6})^2 + (x_1 x_2 - 2)^2$$

$$\mathbf{X}_1 = \begin{Bmatrix} 1 \\ 1 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 10^6 \\ 2 \times 10^{-6} \end{Bmatrix}$$

$$f_1 \approx 10^{12}, \quad f^* = 0.0$$

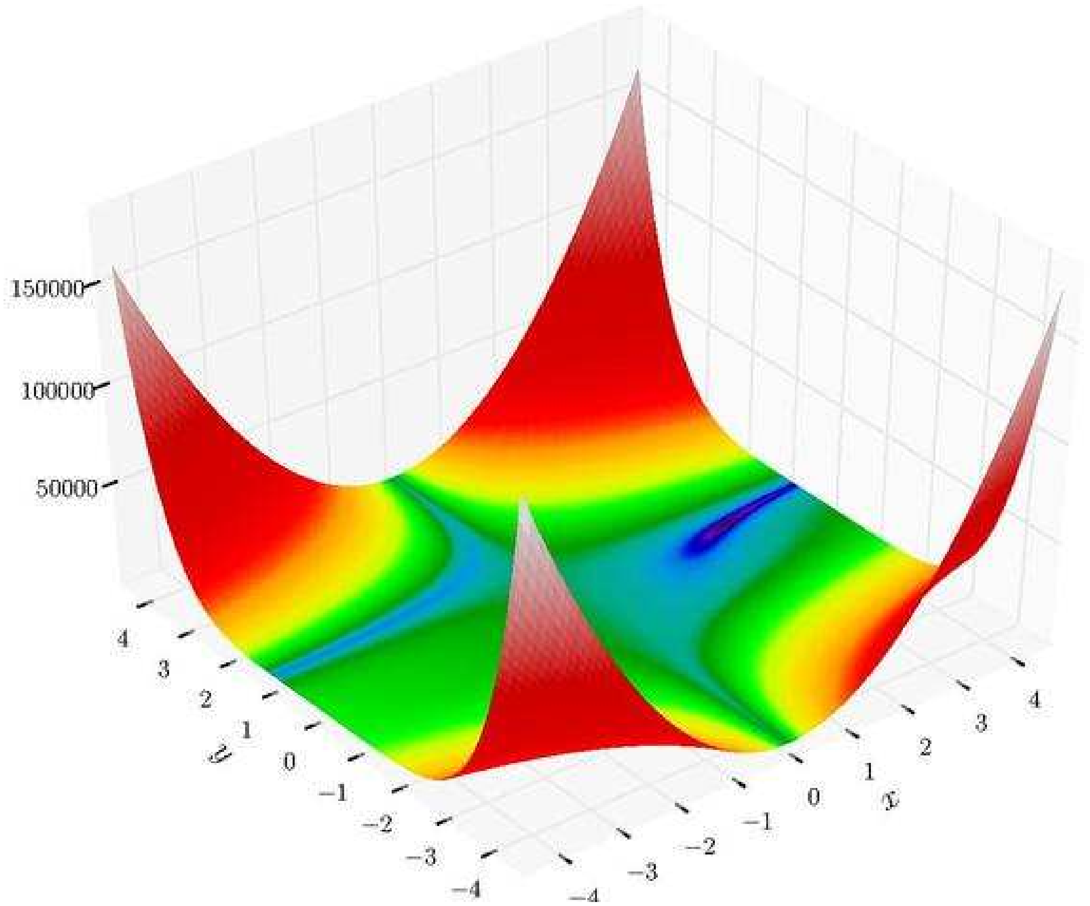


9. Beale's function [6.29]:

$$f(x_1, x_2) = [1.5 - x_1(1 - x_2)]^2 + [2.25 - x_1(1 - x_2^2)]^2 + [2.625 - x_1(1 - x_2^3)]^2$$

$$\mathbf{X}_1 = \begin{Bmatrix} 1 \\ 1 \end{Bmatrix}, \quad \mathbf{X}^* = \begin{Bmatrix} 3 \\ 0.5 \end{Bmatrix}$$

$$f_1 = 14.203125, \quad f^* = 0.0$$



150000
100000
50000
4
3
2
1
0
−1
−2
−3
−4
y
x