Chapter 3

Functions of Several Variables

Let $f(\mathbf{x}) : R^N \mapsto R$ be twice differentiable which will be minimized (or maximized).

- $\mathbf{x}$: design variable of dimension N
- f: objective function
- ∇f: gradient[1]

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial x_3}, \ldots, \frac{\partial f}{\partial x_N} \right)^T$$

3.1 Optimality Criteria

Optimality criteria are necessary

1. to recognize solutions, and
2. they provide motivation for most of the useful search methods

Taylor series expansion

$$\begin{aligned} f(\mathbf{x}) &= f(\overline{\mathbf{x}}) + \mathbf{g}(\overline{\mathbf{x}})\Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{H}(\overline{\mathbf{x}})\Delta\mathbf{x} + O(\|\Delta\mathbf{x}\|^3) \\ &= f(\overline{x}_1, \overline{x}_2) + \left[\frac{\partial f}{\partial x_1}(x_1 - \overline{x}_1) + \frac{\partial f}{\partial x_2}(x_2 - \overline{x}_2) \right] \\ &\quad + \left[\frac{\partial^2 f}{\partial x_1^2}(x_1 - \overline{x}_1)^2 + \frac{\partial^2 f}{\partial x_1 \partial x_2}(x_1 - \overline{x}_1)(x_2 - \overline{x}_2) + \frac{\partial^2 f}{\partial x_2^2}(x_2 - \overline{x}_2)^2 \right] \end{aligned}$$

where all the partial derivatives are evaluated at $\mathbf{x} = \overline{\mathbf{x}} = (\overline{x}_1, \overline{x}_2)$

- $\overline{\mathbf{x}}$ = the current or expansion point in R^N
- $\Delta\mathbf{x} = \mathbf{x} - \overline{\mathbf{x}}$ = the change in $\mathbf{x}$

[1] We need brief discussion on continuity and differentiability of f

- $\mathbf{g}(\overline{\mathbf{x}}) = \nabla f(\overline{\mathbf{x}})$ = the N-component column vector of first derivatives of $f(\mathbf{x})$ evaluated at $\overline{\mathbf{x}}$
- $\mathbf{H}(\overline{\mathbf{x}}) = \nabla^2 f(\overline{\mathbf{x}})$ = the $N \times N$ symmetric matrix of second partial derivatives of $f(\mathbf{x})$ evaluated at $\overline{\mathbf{x}}$, often called the *Hessian* matrix. The element in the i-th row and j-th column is $\dfrac{\partial^2 f}{\partial x_i \partial x_j}$.

$$\mathbf{H} = \mathbf{H}_f = \begin{pmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \cdots & \dfrac{\partial^2 f}{\partial x_1 \partial x_N} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \dfrac{\partial^2 f}{\partial x_N^2} \end{pmatrix} = \mathbf{H}_f^T$$

- $O(\|\Delta\mathbf{x}\|^3)$ = all terms of order greater than or equal to 3 in $\Delta\mathbf{x}$

Neglecting the third and higher order terms yields

$$\Delta f(\mathbf{x}) = f(\mathbf{x}) - f(\overline{\mathbf{x}}) = \mathbf{g}(\overline{\mathbf{x}})\Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{H}(\overline{\mathbf{x}})\Delta\mathbf{x} \tag{3.1}$$

Minimum, maximum, and saddle point

$$\Delta f(\mathbf{x}) = f(\mathbf{x}) - f(\overline{\mathbf{x}}) \geq 0 \tag{3.2}$$

- The point $\overline{\mathbf{x}}$ is a *global* minimum if Eq. (3.2) holds for *all* $\mathbf{x} \in R^N$, and we give this point the symbol $\mathbf{x}^{**}$.
- When Eq. (3.2) holds for some δ-neighborhood that is, for *all* $\mathbf{x}$ such that $\|\mathbf{x} - \overline{\mathbf{x}}\| \leq \delta$ for some $\delta > 0$, then $\overline{\mathbf{x}}$ is a *local* minimum or $\mathbf{x}^*$.
- When

$$\Delta f(\mathbf{x}) = f(\mathbf{x}) - f(\overline{\mathbf{x}}) \leq 0 \tag{3.3}$$

  then $\overline{\mathbf{x}}$ is a maximum.
- Removal of '=' in Eqs. (3.2) and (3.3) produces *strict* minimum and maximum points.
- When Δf is either positive, negative, or zero depending on the choice of nearby points in a δ-neighborhood, then $\overline{\mathbf{x}}$ is a *saddlepoint*

A rough statement of optimality condition In order that the sign of Δf be known for arbitrary values of $\Delta\mathbf{x}$, $\mathbf{g}(\overline{\mathbf{x}})$ must be zero; that is, $\overline{\mathbf{x}}$ must be a *stationary* point. Otherwise, we could force Δf to be plus or minus depending on the sign of $\mathbf{g}(\overline{\mathbf{x}})$ and Δx. Accordingly, $\overline{\mathbf{x}}$ must satisfy the *stationary condition*:

$$\mathbf{g}(\overline{\mathbf{x}}) = 0$$

so Eq. (3.1) becomes

$$\Delta f(\mathbf{x}) = \frac{1}{2} \underbrace{\Delta \mathbf{x}^T \mathbf{H}(\overline{\mathbf{x}}) \Delta \mathbf{x}}_{Q(\Delta \mathbf{x})}$$

The sign of $\Delta f(\mathbf{x})$ depends on the nature of the quadratic form

$$Q(\mathbf{x}) = \mathbf{x}^T \mathbf{H}(\overline{\mathbf{x}}) \mathbf{x}$$

Classification of stationary point The stationary point $\overline{\mathbf{x}}$ is a (See Appendix §A.3.)

- minimum, if $\mathbf{H}(\overline{\mathbf{x}})$ is positive definite.
- maximum, if $\mathbf{H}(\overline{\mathbf{x}})$ is negative definite.
- saddlepoint, if $\mathbf{H}(\overline{\mathbf{x}})$ is indefinite.

Theorem 3.1 (Necessary Conditions) *For $\mathbf{x}^*$ to be a local minimum, it is necessary that*

- $\mathbf{g}(\mathbf{x}^*) = 0$, *and*
- $\mathbf{H}(\mathbf{x}^*)$ *is positive semidefinite.*

Theorem 3.2 (Sufficient Conditions) *If*

- $\mathbf{g}(\mathbf{x}^*) = 0$ *and*
- $\mathbf{H}(\mathbf{x}^*)$ *is positive definite.*

then $\mathbf{x}^*$ is an isolated local minimum of $f(\mathbf{x})$.

Example 3.1 *Find all the stationary points of*

$$f(\mathbf{x}) = 2{x_1}^2 + 4x_1{x_2}^3 - 10x_1x_2 + {x_2}^2$$

and identify the nature of each point.

$$\nabla f(\mathbf{x}) = \begin{pmatrix} 4x_1 + 4{x_2}^3 - 10x_2 \\ 12x_1{x_2}^2 - 10x_1 + 2x_2 \end{pmatrix} = 0$$

$$\nabla^2 f(\mathbf{x}) = \begin{pmatrix} 4 & 12{x_2}^2 - 10 \\ 12{x_2}^2 - 10 & 24x_1x_2 + 2 \end{pmatrix}$$

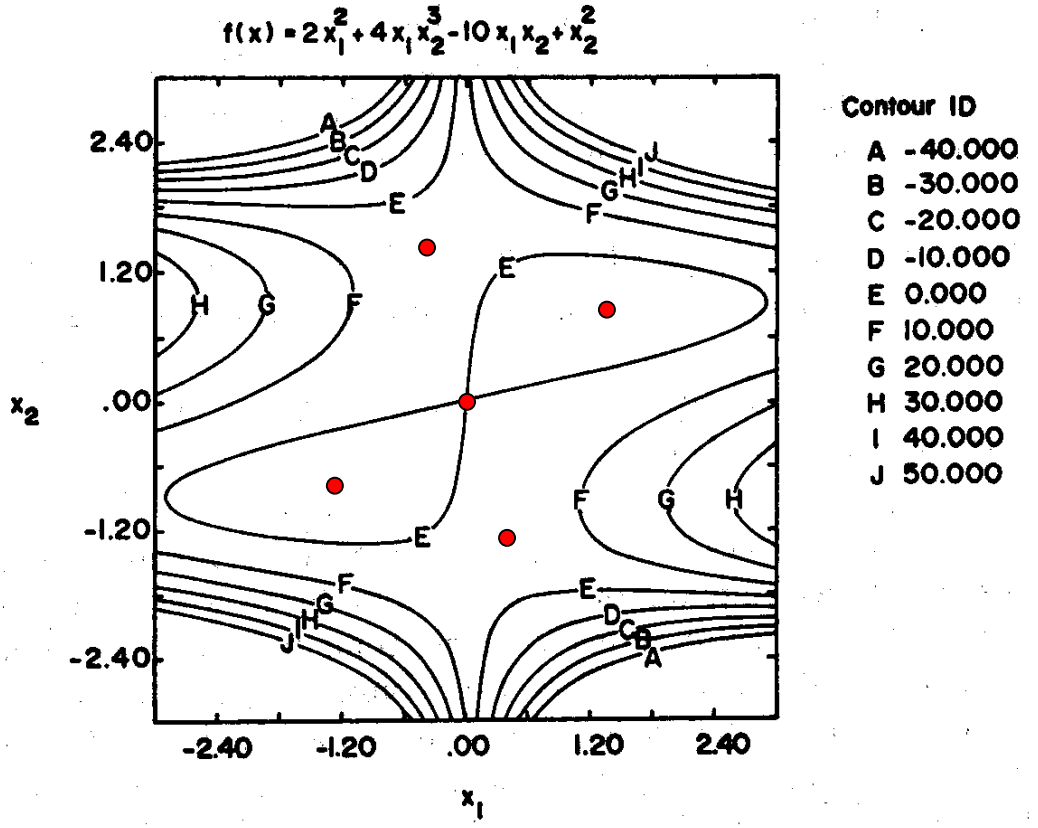

Figure 3.1: Two-variable nonlinear function

Table 3.1: Example 3.1

point	eigenvalues	nature
$(0, 0)$	-7.04988, 13.0499	saddle point
$(-1.5138, -0.85948)$	3.95595, 33.2699	minimum
$(1.5138, 0.85948)$	3.95595, 33.2699	minimum
$(0.153434, -1.61078)$	-21.4471, 21.5545	saddle point
$(-0.153434, 1.61078)$	-21.4471, 21.5545	saddle point

3.2 Direct Search Methods

Optimization techniques for unconstrained multivariable problem

- Direct search methods, which use only function values.
  1. simplex[2] search or S[2]
  2. Hooke-Jeeves pattern search
  3. Powell's conjugate direction
- Gradient methods, which requires accurate values of the first derivative of $f(\mathbf{x})$
  - Cauchy's (or steepest descent, simple gradient)
  - conjugate gradient
  - quasi-Newton (or variable metric)
- Second-order methods, which, in addition to the above, also use the second derivative of $f(x)$
  - Newton(-Raphson)
  - modified Newton
  - Marquardt

3.2.1 The S[2] or Simplex Search Method

In N dimensions, a regular simplex is a polyhedron composed of $N+1$ equidistant points, which form vertices. For example, an equilateral triangle is a simplex in two dimensions; a tetrahedron is a simplex in three dimensions.

[2]It has *no* relationship to the simplex method of LP. The similarity in name is indeed unfortunate.

Simplex search procedures

1. Given an equilateral polyhedron with $N+1$ vertices, $\mathbf{x}_0, \ldots, \mathbf{x}_N$, choose $\mathbf{x}_j$ such that
$$f(\mathbf{x}_j) \geq f(\mathbf{x}_i) \qquad i = 1, \ldots, N+1 \qquad \text{("worst" vertex)}$$

2. Calculate the centroid of the remaining N points
$$\mathbf{x}_c = \frac{1}{N} \sum_{i=0, i \neq j}^{N} x_i$$

3. Calculate the reflection point
$$\mathbf{x}_j^{\text{new}} = \mathbf{x}_j + \lambda(\mathbf{x}_c - \mathbf{x}_j)$$
where $\lambda = 2$.

Troubleshooting and Termination

1. **(Minimum Straddled)** If the selected "worst vertex" was generated in the previous iteration, then choose instead the vertex with the next worst (highest) function value.

2. **(Cycling)** If a given vertex remains unchanged for more than $M = 1.65N + 0.05N^2$ iterations, reduce the size of the simplex by some factor.

3. **(The Termination Criterion)** The search is terminated when the simplex gets small enough or else if the standard deviation of the function values at the vertices gets small enough.

3.2.2 The Hooke-Jeeves Pattern Search Method

Exploratory Move Given a specified step size, which may be different for each coordinate direction and change during the search, the exploration proceeds from an initial point by the specified step size in each coordinate direction. If the function value does not increase, the step is considered successful. Otherwise, the step is retracted and replaced by a step in the opposite direction, which in turn is retained depending upon whether it succeeds or fails. When all N coordinates have been investigated, the exploratory move is completed. The resulting point is termed a *base point*.

Pattern Move A pattern move consists of a single step from the present base point along the line from the previous to the current base point. Thus a new pattern point is calculated as
$$\mathbf{x}_{k+1} = \mathbf{x}_k + (\mathbf{x}_k - \mathbf{x}_{k-1})$$
$\mathbf{x}_{k+1}$ is accepted only when the objective function value is improved.

3.2.3 Powell's Conjugate Direction Method

Quadratic Model

1. It is the simplest type of nonlinear function to minimize, and hence any general technique must work well on a quadrature if it is to have any success with a general function.

2. Near the optimum, all nonlinear function can be approximated by a quadratic. Hence, the behavior of the algorithm on the quadratic will give some indication of how the algorithm will converge for general function.

The motivation of the algorithm stems from the observation that if a quadratic function in N variables

$$q(\mathbf{x}) = a + \mathbf{b}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{C}\mathbf{x}$$

can be transformed so that it is just the sum of perfect squares, then the optimum can be found after exactly N single-variable searches, one with respect to each of the transformed variables. The quadratic term in $q(\mathbf{x})$, namely,

$$Q(\mathbf{x}) = \mathbf{x}^T\mathbf{C}\mathbf{x}$$

with transformation $\mathbf{x} = \mathbf{T}\mathbf{z}$ will yield

$$Q(\mathbf{x}) = \mathbf{z}^T\mathbf{T}^T\mathbf{C}\mathbf{T}\mathbf{z} = \mathbf{z}^T\mathbf{D}\mathbf{z}$$

where $\mathbf{D}$ is a diagonal matrix.

Let $\mathbf{t}_j$ be the jth column of $\mathbf{T}$, then

$$\mathbf{x} = \mathbf{T}\mathbf{z} = \mathbf{t}_1 z_1 + \mathbf{t}_2 z_2 + \cdots + \mathbf{t}_N z_N$$

If a suitable set of transforming vectors $\mathbf{t}_j$; $j = 1, \ldots, N$, conventionally called *conjugate directions*, can be obtained, then the optimum of a quadratic function can be found exactly N single-variable searches.

For given $\mathbf{C}$ the transform $\mathbf{T}$, or its inverse $\mathbf{P}$, where $\mathbf{C} = \mathbf{P}^T\mathbf{D}\mathbf{P}$ can be readily found by the method of "completing the square" [19, §A.4.2]. However, an estimate of $\mathbf{C}$ is not available in our case, because we are seeking to develop a method for optimizing $f(\mathbf{x})$ that uses *only function values*, not first derivatives and certainly not second derivatives.

Parallel Subspace Property Given a quadratic function $q(\mathbf{x})$, two arbitrary but distinct points $\mathbf{x}_1$ and $\mathbf{x}_2$, and a direction $\mathbf{d}$; if $\mathbf{y}_1$ is the solution to

$$q(\mathbf{y}_1) = \min q(\mathbf{x}_1 + \lambda\mathbf{d})$$

and $\mathbf{y}_2$ is the solution to

$$q(\mathbf{y}_2) = \min q(\mathbf{x}_2 + \lambda\mathbf{d})$$

then the direction $(\mathbf{y}_2 - \mathbf{y}_1)$ is $\mathbf{C}$-conjugate to $\mathbf{d}$.

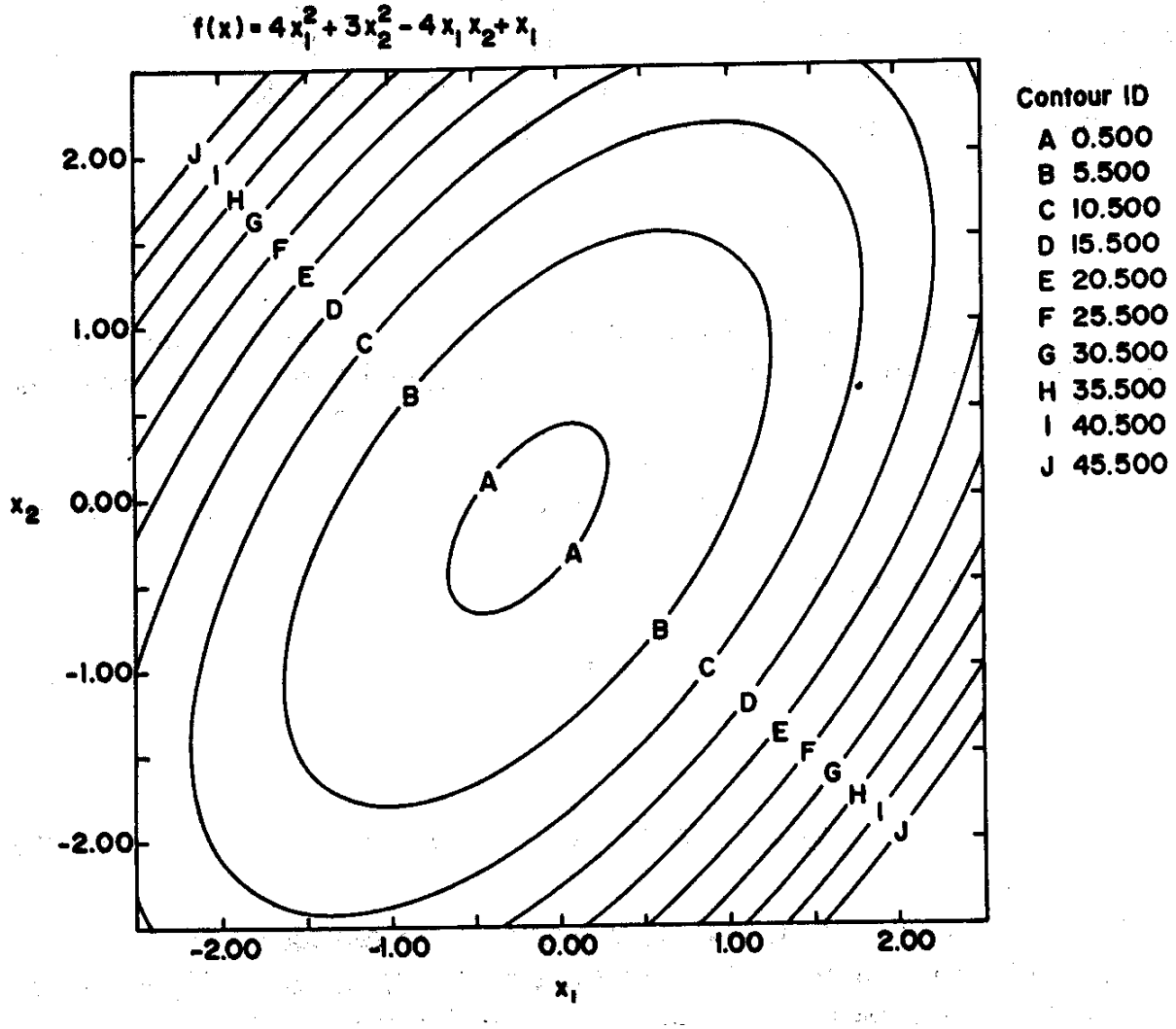

Figure 3.2: A quadratic with cross term

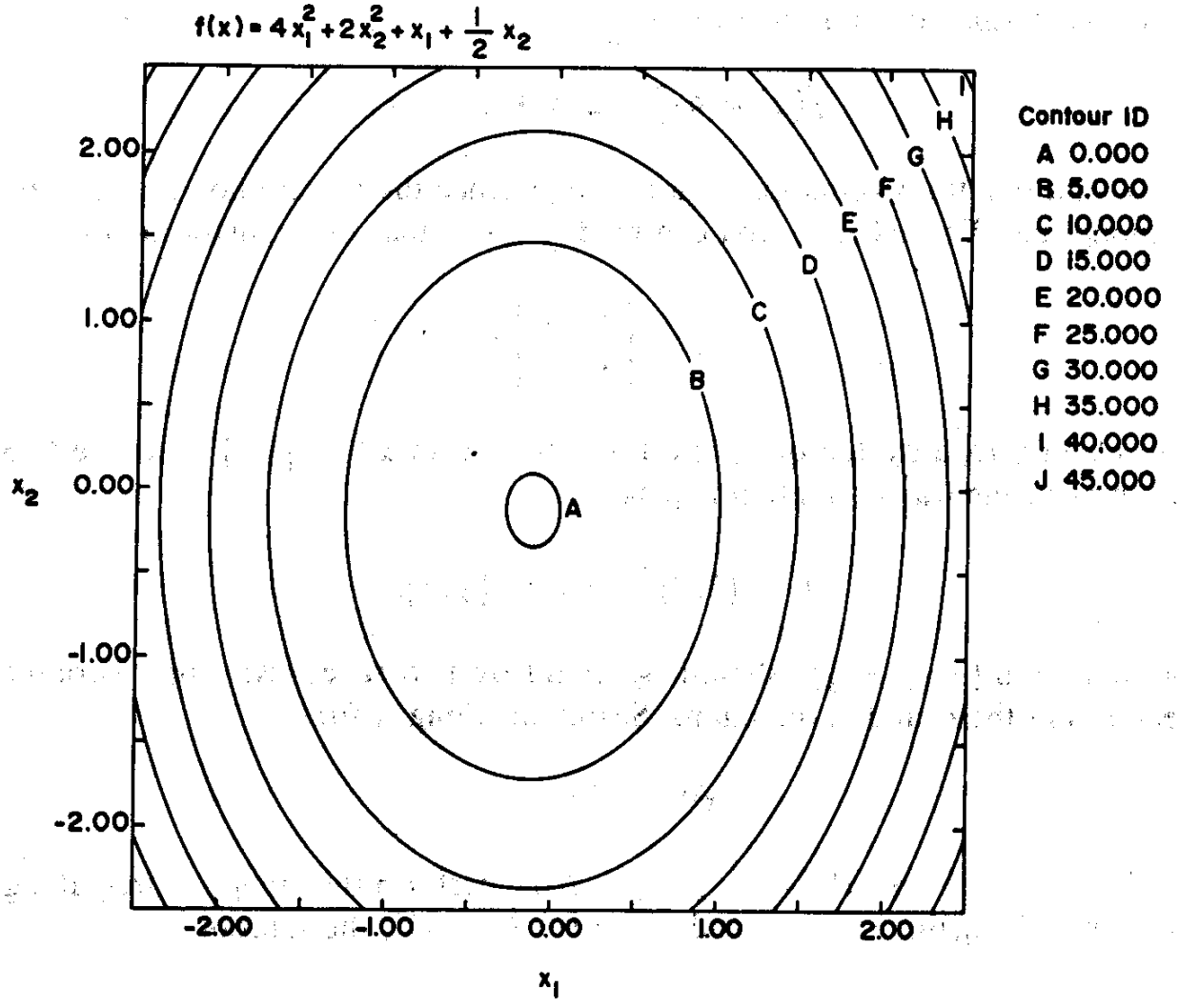

Figure 3.3: A quadratic without cross terms

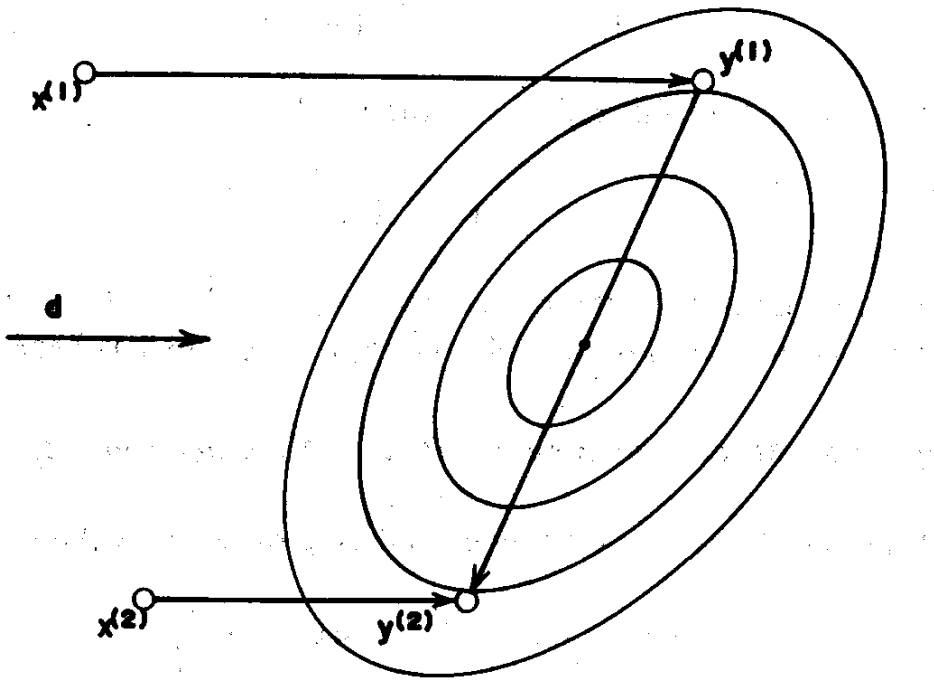

Figure 3.4: Conjugacy in two directions

Definition 3.1 (Conjugate Directions) *Given an $N \times N$ symmetric matrix $\mathbf{C}$, the directions $\mathbf{s}_1,\ \mathbf{s}_2,\ \ldots,\ \mathbf{s}^{(r)}$; $r \leq N$ are said to be* $\mathbf{C}$-conjugate *if the directions are linearly independent, and*

$$\mathbf{s}_i^T \mathbf{C} \mathbf{s}_j = 0$$

for all $i \neq j$

3.3 Gradient-Based Methods

All the methods considered here employ a similar iteration procedure:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{s}(\mathbf{x}_k)$$

where

- $\mathbf{x}_k$ = current estimate of $\mathbf{x}^*$, the solution
- $\mathbf{s}(\mathbf{x}_k) = \mathbf{s}_k$ = search direction in the N space of the design variables x_i; $i = 1, \ldots, N$
- α_k = step-length parameter

3.3.1 First-Derivative Methods

3.3.1.1 Cauchy's Method

Also referred as the *steepest descent* method or the *simple gradient* method

$$f(\mathbf{x}) = f(\overline{\mathbf{x}}) + \mathbf{g}(\overline{\mathbf{x}})^T \Delta\mathbf{x} + \cdots$$

Search direction

$$\mathbf{s}(\overline{\mathbf{x}}) = -\mathbf{g}(\overline{\mathbf{x}})$$

is the direction of most local descent or the steepest descent direction.

Line search For $\mathbf{x} = \overline{\mathbf{x}} + \alpha\mathbf{s}(\overline{\mathbf{x}})$ or $x_1 = \overline{x}_1 + \alpha s_1$ and $x_2 = \overline{x}_2 + \alpha s_2$,

$$\min_{\alpha > 0} F(\alpha) = f(\mathbf{x}) = f(\overline{\mathbf{x}} + \alpha\mathbf{s}) = f(x_1 + \alpha s_1, x_2 + \alpha s_2)$$

$$F'(\alpha) = \frac{\partial f}{\partial x_1}\frac{dx_1}{d\alpha} + \frac{\partial f}{\partial x_2}\frac{dx_2}{d\alpha} = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}\right)\begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = -\mathbf{g}^T(\mathbf{x})\mathbf{g}(\overline{\mathbf{x}}) = 0$$

Descent property

$$\mathbf{x} - \overline{\mathbf{x}} = \alpha\mathbf{s}(\overline{\mathbf{x}}) = -\alpha\mathbf{g}(\overline{\mathbf{x}})$$

and

$$f(\mathbf{x}) = f(\overline{\mathbf{x}}) + \mathbf{g}(\overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}}) + \cdots$$
$$f(\mathbf{x}) - f(\overline{\mathbf{x}}) = -\alpha\mathbf{g}(\overline{\mathbf{x}})\mathbf{g}(\overline{\mathbf{x}}) + \cdots < 0$$

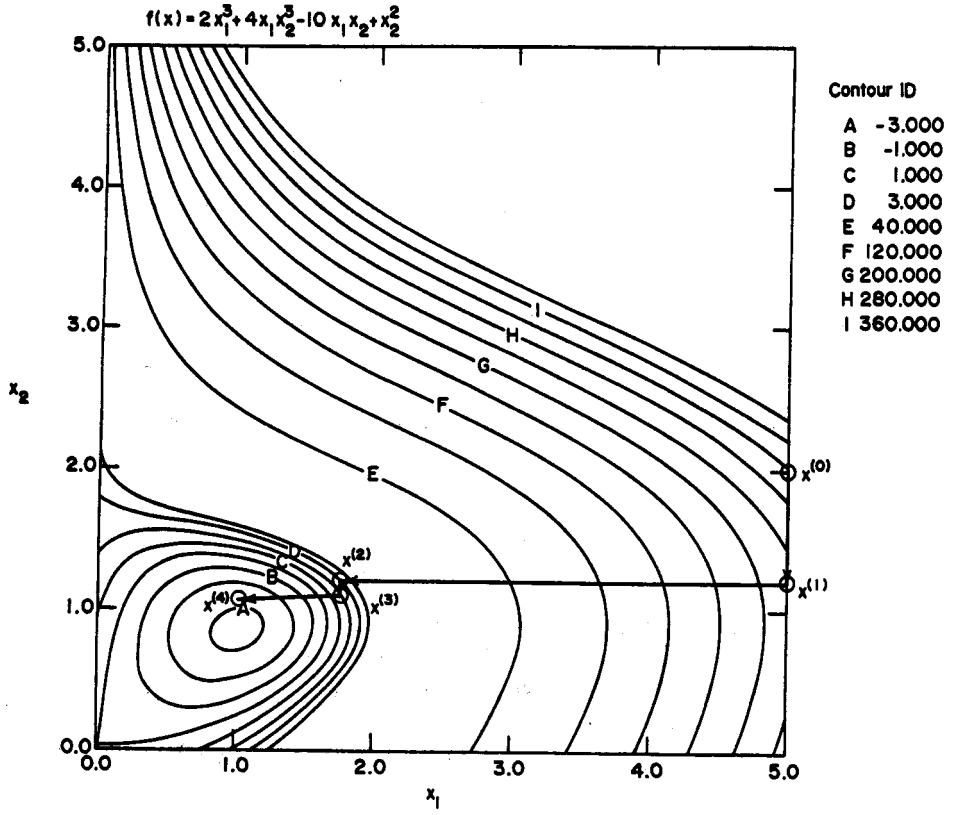

Figure 3.5: Powell's conjugate direction method

3.3.1.2 Conjugate Gradient Methods

Fletcher-Reeves

$$\mathbf{s}_k = -\mathbf{g}_k + \frac{\|\mathbf{g}_k\|^2}{\|\mathbf{g}_{k-1}\|^2}\mathbf{s}_{k-1}$$

where $\mathbf{g}_k = \mathbf{g}(\mathbf{x}_k)$

Polak-Ribiere

$$\mathbf{s}_k = -\mathbf{g}_k + \frac{(\mathbf{g}_k - \mathbf{g}_{k-1})^T\mathbf{g}_k}{\|\mathbf{g}_{k-1}\|^2}\mathbf{s}_{k-1}$$

3.3.1.3 Quasi-Newton Methods

$$\mathbf{s}_k = -\mathbf{A}_k\mathbf{g}_k$$

where $\mathbf{A}_k$ is an $N \times N$ matrix called the *metric*. Methods that employ directions of this form are often called *variable metric* methods, because $\mathbf{A}$ changes at each iteration.

Davidon-Fletcher-Powell (DFP)

$$\mathbf{A}_k = \mathbf{A}_{k-1} + \frac{\Delta\mathbf{x}_{k-1}\Delta\mathbf{x}_{k-1}^T}{\Delta\mathbf{x}_{k-1}^T\Delta\mathbf{g}_{k-1}} - \frac{\mathbf{A}_{k-1}\Delta\mathbf{g}_{k-1}\Delta\mathbf{g}^{(k-1)T}\mathbf{A}_{k-1}}{\Delta\mathbf{g}^{(k-1)T}\mathbf{A}_{k-1}\Delta\mathbf{g}_{k-1}}$$

Broyden-Fletcher-Shanno (BFS) or BFGS[3]

$$\mathbf{A}_{k+1} = \left[\mathbf{I} - \frac{\Delta\mathbf{x}_k\Delta\mathbf{g}_k^T}{\Delta\mathbf{x}_k^T\Delta\mathbf{g}_k}\right]\mathbf{A}_k\left[\mathbf{I} - \frac{\Delta\mathbf{x}_k\Delta\mathbf{g}_k^T}{\Delta\mathbf{x}_k^T\Delta\mathbf{g}_k}\right] + \frac{\Delta\mathbf{x}_k\Delta\mathbf{x}_k^T}{\Delta\mathbf{x}_k^T\Delta\mathbf{g}_k}$$

3.3.2 Second-Derivative Methods

3.3.2.1 Newton's Method

$$f(\mathbf{x}) = f(\overline{\mathbf{x}}) + \mathbf{g}(\overline{\mathbf{x}})\Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{H}(\overline{\mathbf{x}})\Delta\mathbf{x} + O(\|\Delta\mathbf{x}\|^3)$$

quadratic approximation

$$\tilde{f}(\mathbf{x};\mathbf{x}_k) = f(\mathbf{x}_k) + \mathbf{g}(\mathbf{x}_k)\Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T\mathbf{H}(\mathbf{x}_k)\Delta\mathbf{x}$$

$$\nabla\tilde{f}(\mathbf{x};\mathbf{x}_k) = \mathbf{g}(\mathbf{x}_k) + \mathbf{H}(\mathbf{x}_k)\Delta\mathbf{x} = 0$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - [\mathbf{H}(\mathbf{x}_k)]^{-1}\mathbf{g}(\mathbf{x}_k)$$

[3]Broyden-Fletcher-Goldfarb-Shanno

3.3.2.2 Modified Newton's method

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k[\mathbf{H}(\mathbf{x}_k)]^{-1}\mathbf{g}(\mathbf{x}_k)$$

- choose α_k that minimizes $f(\mathbf{x}_{k+1})$
- ensures descent property

3.3.2.3 Marquardt's method

Cauchy + Newton

$$\mathbf{s}(\mathbf{x}_k) = -[\mathbf{H}_k + \lambda_k\mathbf{I}]^{-1}\mathbf{g}(\mathbf{x}_k)$$

3.3.3 A Gradient-Based Algorithm

- Calculate $\mathbf{s}$
- Find α (> 0) such that $f(\mathbf{x} + \alpha\mathbf{s})$ is minimum

Conjugate gradient and quasi-Newton family are the most popular unconstrained minimization algorithms. These algorithms are continuously evoling and widely available in both commercial and public domain. The algorithms inherently have triple iteration hierarchy, and the algorithm providers use their own way of counting the number of iterations. The innermost loop is the *line search iteration*, then the *conjugate direction iteration*, and the outermost one.

Line search iteration Let $F(\alpha) = f(\mathbf{x}_k + \alpha\mathbf{s}_k)$ then the line search is given by

$$\min_{\alpha>0} F(\alpha)$$

Using chain rule,

$$F'(\alpha) = \mathbf{g}^T(\mathbf{x}_k + \alpha\mathbf{s}_k)\mathbf{s}_k$$

$F'(\alpha)$ will be zero at the point where the search direction vector, $\mathbf{s}_k$, lies on the tangent of the contour of $f(\mathbf{x}) = f_k$. which provides a very useful termination criterion for the line search. A typical termination criterion for line search of gradient-based algorithm is

$$\frac{F'(\alpha_k)}{F'(0)} = \frac{\bar{\mathbf{g}}^T\mathbf{g}_k}{\mathbf{g}_k^T\mathbf{g}_k} \leq \epsilon$$

where $\bar{\mathbf{g}} = \mathbf{g}^T(\mathbf{x}_k + \bar{\alpha}\mathbf{s}_k)$. Let

$$\beta(\bar{\alpha}) = \frac{\bar{\mathbf{g}}^T\mathbf{g}_k}{\mathbf{g}_k^T\mathbf{g}_k}$$

then $\beta(0) = 1$ which corresponds to the current base point $\mathbf{x}_k$.

Assume that f is quadratic so that $\mathbf{g}(\mathbf{x}) = 2\mathbf{C}\mathbf{x} + \mathbf{b}$. We claim that

$$\alpha^* = \frac{\bar{\alpha}}{1-\beta}$$

is the minimum point of $F(\alpha)$, i. e., $F'(\alpha^*) = 0$. Note that $\mathbf{g}_k = 2\mathbf{C}\mathbf{x}_k + \mathbf{b}$ and

$$\bar{\mathbf{g}} = 2\mathbf{C}(\mathbf{x}_k + \bar{\alpha}\mathbf{s}) + \mathbf{b} = \mathbf{g}_k + 2\bar{\alpha}\mathbf{C}\mathbf{s}$$

It can be easily shown from

$$\begin{aligned} \frac{\mathbf{g}_k^T\mathbf{g}_k + 2\bar{\alpha}\mathbf{g}_k^T\mathbf{C}\mathbf{s}_k}{\mathbf{g}_k^T\mathbf{g}_k} = 1 + 2\bar{\alpha}\frac{\mathbf{g}_k^T\mathbf{C}\mathbf{s}_k}{\mathbf{g}_k^T\mathbf{g}_k} &= \beta \\ \frac{\mathbf{g}_k^T\mathbf{g}_k + 2\alpha^*\mathbf{g}_k^T\mathbf{C}\mathbf{s}_k}{\mathbf{g}_k^T\mathbf{g}_k} = 1 + 2\alpha^*\frac{\mathbf{g}_k^T\mathbf{C}\mathbf{s}_k}{\mathbf{g}_k^T\mathbf{g}_k} &= 0 \end{aligned}$$

It enhances the performance of the line search, especially when the objective function resembles a quadratic function, and allows the tolerance ϵ be typically 0.1, and even 0.9 for some special circumstances.

Conjugacy iteration As we already discussed, the sequence of search direction $\mathbf{s}_k$'s, $k = 1, \ldots,$ are conjugate directions with respect to the Hessian matrix of $f(\mathbf{x})$ if

- $f(\mathbf{x})$ is quadratic, and (C-1)
- the line search is exact. (C-2)

The search directions are

- linearly independent to each other, and (P-1)
- descent direction, i. e., $(\mathbf{s}_k)^T\mathbf{g}_k < 0$. (P-2)

In this case, we expect that there exist N conjugate directions, and the optimization iteration should terminate after N search-direction generation — After N-th line search, $||\mathbf{g}(\mathbf{x}_N)|| = 0$. In practice, neither (C-1) nor (C-2) is satisfied. Hence $||\mathbf{g}|| > \epsilon$ where ϵ is the allowable error limit. If we generate $(N+1)$-st search direction, then it will be no more conjugate to the previously generated search direction, since (P-1). It requires to generate new set of conjugate search directions:

$$\mathbf{s}_{N+1} = -\mathbf{g}(\mathbf{x}_{N+1})$$

In fact, the first N search directions, $\mathbf{s}_1, \ldots, \mathbf{s}_N$ do not form an *exact* conjugate set if either (C-1) or (C-2) is violated. Hence, in some cases, (P-2) is violated:

$$(\mathbf{s}_k)^T\mathbf{g}_k > 0 \tag{3.4}$$

The loss of conjugacy is not rare if N is large, and Eq. (3.4) is another termination criteria of conjugacy iteration.

Outermost iteration The iteration count of outermost loop increased by 1 whenever the conjugate direction set restarts. The single outermost iteration might be compared to the single classical Newton-Raphson iteration in some sense. But we expect that the conjugate gradient or quasi-Newton methods are computationally more efficient, especially when the objective function is complicated and/or has large number of decision variables, if we consider the effort required to complete single outermost iteration.

Termination Criteria The optimization iteration terminate in any loop if a prespecified combination of following criteria is satisfied.

- $||\mathbf{x}_{k+1} - \mathbf{x}_k|| \leq \epsilon$ or $\dfrac{||\mathbf{x}_{k+1} - \mathbf{x}_k||}{||\mathbf{x}_k||} \leq \epsilon$
- $|f_{k+1} - f_k| \leq \epsilon$
- $||\mathbf{g}_k|| \leq \epsilon$
- Maximum number of function evaluation

Conjugate gradient and quasi-Newton algorithms

1. Step 1: Initialize conjugate direction set.
   - Set $i = 1$ and $\Phi(i-1, \mathbf{s}, \mathbf{x}, \mathbf{g}) = 0$
2. Step 2: Generation of conjugate direction.
   - Compute $f_i = f(\mathbf{x}_i)$ and $\mathbf{g}_i = \mathbf{g}(\mathbf{x}_i)$
   - $\mathbf{s}_i = -\mathbf{g}_i + \Phi(i-1, \mathbf{s}, \mathbf{x}, \mathbf{g})$
   - If $\mathbf{s}_i^T \mathbf{g}_i > 0$ then go to Step 1. (Loss of conjugacy terminates the conjugacy iteration prematurely.)
3. Step 3: Line search
   - $\min\limits_{\alpha>0} f(\mathbf{x}_i + \alpha \mathbf{s}_i)$
   - $\mathbf{x}^{(i+1)} = \mathbf{x}_i + \alpha_i \mathbf{s}_i$
   - Set $i = i + 1$
4. Step 4
   - If $i > N$ then go to Step 1 (Outermost loop), otherwise go to Step 2 (Conjugacy loop).

3.3.4 Numerical Gradient Approximations

Forward difference

$$\left.\frac{\partial f(\mathbf{x})}{\partial x_i}\right|_{\mathbf{x}=\overline{\mathbf{x}}} = \frac{f(\overline{\mathbf{x}}+\varepsilon\mathbf{e}_i)-f(\overline{\mathbf{x}})}{\varepsilon} \quad (90)$$

Central difference

$$\left.\frac{\partial f(\mathbf{x})}{\partial x_i}\right|_{\mathbf{x}=\overline{\mathbf{x}}} = \frac{f(\overline{\mathbf{x}}+\varepsilon\mathbf{e}_i)-f(\overline{\mathbf{x}}-\varepsilon\mathbf{e}_i)}{2\varepsilon} \quad (91)$$

3.4 Comparison of Methods and Numerical Results

Test problems

- Himmelblau's function

$$f(\mathbf{x}) = 2{x_1}^2 + 4x_1{x_2}^3 - 10x_1x_2 + {x_2}^2$$

- Rosenbrock's function

$$f(\mathbf{x}) = 100({x_2} - {x_1}^2)^2 + (1 - x_1)^2$$

- Fenton and Eason's function

$$f(\mathbf{x}) = 1.2 + 0.1{x_1}^2 + \frac{0.1 + 0.1{x_2}^2}{{x_1}^2} + \frac{0.1{x_1}^2{x_2}^2 + 10}{(x_1x_2)^4}$$

- Wood's function

$$\begin{aligned} f(\mathbf{x}) \;=\; & 100({x_2} + {x_1}^2)^2 + (1 - x_1)^2 + 90({x_4} - {x_3}^2)^2 + (1 - x_3)^2 \\ & +10.1\left[(x_2 - 1)^2 + (x_4 - 1)^2\right] + 19.8(x_2 - 1)(x_4 - 1) \end{aligned}$$

3.5 Assignments

3.5.1 Reading Materials

- Reklaitis, *et. al.*'s Chapter 3 [19]
- Edgar & Himmelblau's Chapters 4 & 6 [5]
- Rao's Chapter 6 [17]
- Chong & Żak's Chapters 6 & 8 – 11 [4]

Example 3.2 *Cauchy vs. FR for quadratic function*

$$\min f(\mathbf{x}) = 8x_1^2 + 4x_1x_2 + 5x_2^2$$

Cauchy's method using Mathematica

1. Initialization

```
f = 8 x1^2 + 4 x1 x2 + 5 x2^2
g = {D[f, x1], D[f, x2]}
n = 1
{u1[n], u2[n]} = {-4, -4}
```

2. Cauchy iteration

```
{s1, s2} = -g/.{x1 -> u1[n], x2 -> u2[n]}
m = FindMinimum[f/.{x1 -> u1[n] + a s1, x2 -> u2[n] +a s2}, {a, 0}]
n = n+1
{u1[n], u2[n]} = {u1[n-1], u2[n-1]} + (a/.Part[m, 2]) {s1, s2}
```

3. Graphic display of result

```
p1 = ListPlot[Table[{u1[i], u2[i]}, {i, n}], PlotJoined -> True]
p2 = ContourPlot[f, {x1,-1,1}, {x2,-1,1}, ContourShading -> False]
p3 = ContourPlot[f, {x1,-4,1}, {x2,-4,1}, ContourShading -> False]
Show[p1, p2, p3]
```

Fletcher-Reeves' method using Mathematica

1. Initialization

```
f = 8 x1^2 + 4 x1 x2 + 5 x2^2
g = {D[f, x1], D[f, x2]}
n = 1
{u1[n], u2[n]} = {-4, -4}
sa = {0, 0}
ga = 1
```

2. FR iteration

```
{g1, g2} = g /. {x1 -> u1[n], x2 -> u2[n]}
gb = g1 g1 + g2 g2
{s1, s2} = -{g1, g2} + sa gb /ga
m = FindMinimum[f/.{x1 -> u1[n] + a s1, x2 -> u2[n] + a s2}, {a, 0}]
n = n + 1
{u1[n], u2[n]} = {u1[n-1], u2[n-1]} + (a /. Part[m, 2]) {s1, s2}
ga = gb
sa = {s1, s2}
```

Example 3.3 *Cauchy vs. FR for non-quadratic function*

$$\min f(\mathbf{x}) = 25(x_1 + 2x_2 - 4)^4 + (2x_1 - x_2 - 1)^2$$

Example 3.4 (Example 1.3) *This is a typical example of illustrating the importance of proper scaling of system model.*

Formulation of Least-Squares Problem

$$\min_{a,b} f(a,b) = \sum_{i=1}^{8} [P(v_i, T_i) - P_i]^2 = \sum_{i=1}^{8} \left[\frac{RT_i}{v_i - b} - \frac{a}{\sqrt{T_i} v_i (v_i + b)} - P_i \right]^2 \tag{1.1}$$

- Let the initial guess of (a, b) is $(0, 0)$ then

$$\mathbf{g}(0,0) = \begin{pmatrix} -3.86 \times 10^{-5} \\ 17.4 \end{pmatrix}$$

- According to Reklaitis[19, p. 22], $f(6.377 \times 10^7, 29.7) = 0.097$ is the minimum (RMSE $= 0.11$).

$$\mathbf{g}(6.377 \times 10^7, 29.7) = \begin{pmatrix} 3.47 \times 10^{-8} \\ -0.0422 \end{pmatrix}$$

- According to `FindMinimum` of Mathematica with $(a_0, b_0) = (0, 0)$, $f(6.480 \times 10^7, 31.24) = 0.0852$ is the minimum (RMSE $= 0.103$).

Scaling of system model

- From thermodynamics: $v > b > 0$

$$O(b) = O(v) = \min_i v_i = 400$$

- From thermodynamics: $a > 0$ and $O\left(\frac{RT}{v-b}\right) = O\left(\frac{a}{\sqrt{T} v (v+b)}\right)$

$$O(a) = RT\sqrt{T}v \frac{v+b}{v-b} \approx RT\sqrt{T}v = \min_i RT_i \sqrt{T_i} v_i = 1.48 \times 10^8$$

Defind new decision variables (α, β) so that

$$\begin{aligned} a &= 1.48 \times 10^8 \alpha \\ b &= 400\beta \end{aligned}$$

and

$$P = \frac{RT}{v - 400\beta} - \frac{1.48 \times 10^8 \alpha}{\sqrt{T} v (v + 400\beta)} \text{(b)}$$

Thanks to the proper *scaling*, we will be much happier with the gradient of $f(\alpha, \beta)$:

$$\mathbf{g}(0,0) = \begin{pmatrix} -5711 \\ 6977 \end{pmatrix} \qquad \text{and} \qquad \mathbf{g}(0.4309, 0.0743) = \begin{pmatrix} 5.132 \\ -16.90 \end{pmatrix}$$

`FindMinimum` found the minimum $f(0.4378, 0.0781) = 0.0852$

Matlab m-file `c:\che542\ex0104.m` **for Example 1.4 of Reklaitis**

```
function f=fun(x)

R=82.06; a=1.48e+08*x(1); b=400*x(2);

P(1)=33.0; v(1)=500.0; T(1)=273.0;
P(2)=43.0; v(2)=500.0; T(2)=323.0;
P(3)=45.0; v(3)=600.0; T(3)=373.0;
P(4)=26.0; v(4)=700.0; T(4)=273.0;
P(5)=37.0; v(5)=600.0; T(5)=323.0;
P(6)=39.0; v(6)=700.0; T(6)=373.0;
P(7)=38.0; v(7)=400.0; T(7)=273.0;
P(8)=63.6; v(8)=400.0; T(8)=373.0;

f=0;
for i=1:8,
dP=R*T(i)/(v(i)-b)-a/sqrt(T(i))/v(i)/(v(i)+b)-P(i);
f=f+dP*dP;
end
```

Execution from Matlab command window

```
> x0=[0,0]

x0 =

     0     0

> options(1)=1

options =

     1

> cd c:\che542
> x=fminu('ex0104',x0,options)
f-COUNT   FUNCTION    STEP-SIZE      GRAD/SD  LINE-SEARCH
    4      876.769        0.001   -8.13e+007

Warning: Matrix is close to singular or badly scaled.
         Results may be inaccurate. RCOND = 1.296142e-018

   11      470.706  5.8419e-006   -5.79e+007  incstep
```

```
   17      384.293  0.000153823   -4.69e+004  incstep
   23      8.04672     0.829131        -6.39  incstep
   30       1.7089      14.5935       -0.168  incstep IF
   35      1.11338      0.10918        -4.72  incstep
   41     0.319636     0.843285    0.0000238
   47    0.0874874      1.80015    -0.000915  incstep
   52    0.0852005      1.04197   -3.68e-007  incstep
   57    0.0851855     0.906612     3.5e-008  int_st
Optimization Terminated Successfully
Gradient less than options(2)
 NO OF ITERATIONS=57

x =

    0.4378    0.0781
```

Bibliography

[1] Beveridge, G. S. & R. S. Schechter, 1970, *Optimization: Theory and Practice*, McGraw-Hill.

[2] Biegler, L. T., E. I. Grossmann & A. W. Westerberg, 1997, *Systematic Methods of Chemical Process Design*, Prentice-Hall.

[3] Bryson, Jr., A. E. & Y.-C. Ho, 1975, *Applied Optimal Control: Optimization, Estimation, and Control*, Revised Printing, Hemisphere.

[4] Chong, E. K. P. & S. H. Żak, 1996, *An Introduction to Optimization*, Wiley.

[5] Edgar, T. F. & D. M. Himmelblau, 1989, *Optimization of Chemical Processes*, McGraw-Hill.

[6] Edgeworth, F. Y., 1987, *Mathematical Psychics*, University Microfilms International (Out-of Print Books on Demand, the original edition in 1881).

[7] Floudas, C. A. & P. M. Pardalos (eds.), 2001, *Encyclipedia of Optimization*, Kluwer.

[8] Floudas, C. A., 2000, *Deterministic Global Optimization: Theory, Methods and Applications*, Kluwer.

[9] Floudas, C. A. & P. M. Pardalos (eds.), 2000, *Optimization in Computational Chemistry and Molecular Biology*, Kluwer.

[10] Gelfand, I. M. & S. V. Fomin, 1963, *Calculus of Variation*, Revised English editon translated and edited by R. A. Silverman, Prentice-Hall.

[11] Horst, R. & P. M. Pardolos (eds.), 1995, *Handbook of Global Optimization*, Kluwer.

[12] Horst, R. & H. Tuy, 1996, *Global Optimization: Deterministic Approach*, 3rd ed., Springer.

[13] McCormick, G. P., 1967, "Second order conditions for constrained minima," *SIAM J. Appl. Math.*, **15**(3), 641 – 652.

[14] Miettinen. K. M., 1999, *Nonlinear Multiobjective Optimization*, Kluwer.

[15] Pareto, V., 1964, *Cours d'Economie Politique*, Libraire Droz, Genève (the first edition in 1896).

[16] Pareto, V., 1971, *Manual of Political Economy*, MacMillan Press (the original edition in French in 1927).

[17] Rao, S. S., 1996, *Engineering Optimization: Theory and Practice*, 3rd ed., Wiley.

[18] Ray, W. H., 1981, *Advanced Process Control*, McGraw-Hill.

[19] Reklaitis, G. V., A. Ravindran & K. M. Ragsdell, 1983, *Engineering Optimization: Methods and Applications*, Wiley.

Optimization Journals

- AIAA Journal
- ASCE Journal of Structural Engineering
- ASME Journal of Mechanical Design
- Computers and Chemical Engineering
- Computers and Operations Research
- Computers and Structures
- Engineering Optimization
- International Journal for Numerical Methods in Engineering
- Journal of Optimization Theory and Applications
- Management Sceience
- Operations Research
- SIAM Journal of Optimization
- Structural Optimization