Computer aided ship design

Part 3. Optimization Methods

November 2008

Prof. Kyu-Yeul Lee

Department of Naval Architecture and Ocean Engineering,
Seoul National University of College of Engineering

Advanced
Ship
Design
Automation
Laboratory

5.4 CSD(Constrained Steepest Descent) 방법

Advanced
Ship
Design
Automation
Laboratory

탐색 방향을 결정하기 위한 2차 계획 문제의 정식화(1)

$$Minimize \quad f(\mathbf{x}+\Delta\mathbf{x}) \cong f(\mathbf{x}) + \nabla f^T(\mathbf{x})\Delta\mathbf{x} + 0.5\Delta\mathbf{x}^T\mathbf{H}\Delta\mathbf{x}$$

Taylor 급수의 2차항까지 고려한 목적 함수

$$Subject\ to \quad h_j(\mathbf{x}+\Delta\mathbf{x}) \cong h_j(\mathbf{x}) + \nabla h_j{}^T(\mathbf{x})\Delta\mathbf{x} = 0;\ j = 1\ to\ p$$

Taylor 급수의 1차항(선형항)만 고려한 등호 제약 조건

$$g_j(\mathbf{x}+\Delta\mathbf{x}) \cong g_j(\mathbf{x}) + \nabla g_j{}^T(\mathbf{x})\Delta\mathbf{x} \le 0;\ j = 1\ to\ m$$

Taylor 급수의 1차항(선형항)만 고려한 부등호 제약 조건

여기서, $\bar{f} = f(\mathbf{x}+\Delta\mathbf{x}) - f(\mathbf{x}),\ e_j = -h_j(\mathbf{x}),\ b_j = -g_j(\mathbf{x}),$

$c_i = \partial f(\mathbf{x})/\partial x_i,\ n_{ij} = \partial h_j(\mathbf{x})/\partial x_i,\ a_{ij} = \partial g_j(\mathbf{x})/\partial x_i,$

$d_i = \Delta x_i$ 라고 가정하면

Matrix form

$$Minimize \quad \bar{f} = \mathbf{c}^T{}_{(1\times n)}\mathbf{d}_{(n\times 1)} + \frac{1}{2}\mathbf{d}^T{}_{(1\times n)}\mathbf{H}_{(n\times n)}\mathbf{d}_{(n\times 1)}$$

: 2차 형식의 목적 함수

$$Subject\ to \quad \mathbf{N}^T{}_{(p\times n)}\mathbf{d}_{(n\times 1)} = \mathbf{e}_{(p\times 1)}$$

: 선형화 된 등호 제약 조건

$$\mathbf{A}^T{}_{(m\times n)}\mathbf{d}_{(n\times 1)} \le \mathbf{b}_{(m\times 1)}$$

: 선형화 된 부등호 제약 조건

황금분할법에 의한 이동 거리의 결정

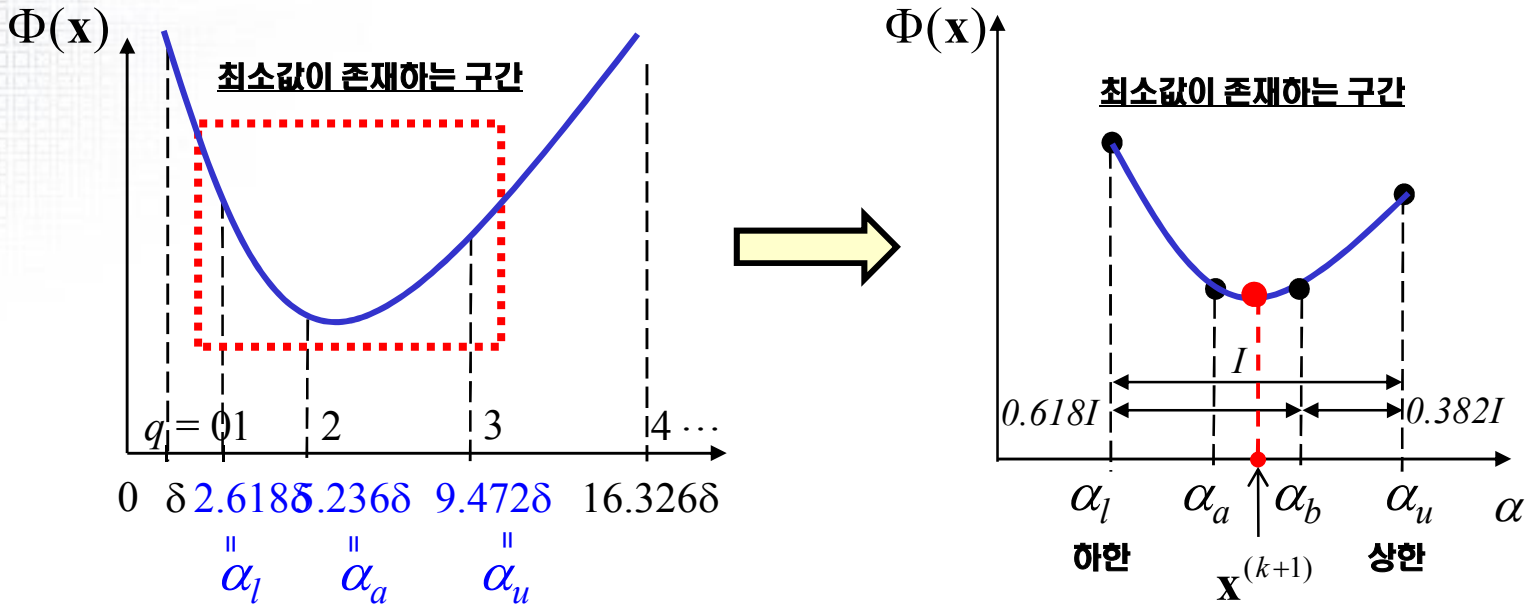
강하 조건을 검토하기 위한 시행 설계점

$\mathbf{x}^{(k,j)} = \mathbf{x}^{(k)} + \alpha_{(k,j)}\mathbf{d}^{(k)}$ ➡ 개선된 설계점을 구하기 위해 $\alpha_{(k,j)}$를 얼마로 해야 하는가?

$\alpha_{(k,j)}$를 변경시켜 가면서 현재의 설계점보다 강하 함수를 감소시키는 설계점을 찾음
(1차원 탐색 방법(예: 황금 분할법) 이용 가능)

1차원 탐색 방법(예: 황금 분할법)을 이용한 개선된 설계점 $\mathbf{x}^{(k+1)}$ 결정
(최종 결정된 $\mathbf{x}^{(k,j)}$ 가 $\mathbf{x}^{(k+1)}$로 변경됨)

황금 분할법에 의해 최소값이 존재하는 구간을 찾은 뒤, 구간을 다시 내분하여 실제 최소값의 x좌표를 구함



2차 계획 문제(Quadratic Programming Problem)의 정식화

$$\textit{Minimize} \quad \bar{f} = \mathbf{c}^T{}_{(1\times n)}\mathbf{d}_{(n\times 1)} + \frac{1}{2}\mathbf{d}^T{}_{(1\times n)}\mathbf{H}_{(n\times n)}\mathbf{d}_{(n\times 1)}$$

$$\textit{Subject to} \quad \mathbf{N}^T{}_{(p\times n)}\mathbf{d}_{(n\times 1)} = \mathbf{e}_{(p\times 1)}$$

$$\mathbf{A}^T{}_{(m\times n)}\mathbf{d}_{(n\times 1)} \le \mathbf{b}_{(m\times 1)}$$

↓ $\mathbf{H}_{(n\times n)} = \mathbf{I}_{(n\times n)}$라고 가정해서 근사화 한다.

$$\textit{Minimize} \quad \bar{f} = \mathbf{c}^T{}_{(1\times n)}\mathbf{d}_{(n\times 1)} + \frac{1}{2}\mathbf{d}^T{}_{(1\times n)}\mathbf{I}_{(n\times n)}\mathbf{d}_{(n\times 1)} = \mathbf{c}^T{}_{(1\times n)}\mathbf{d}_{(n\times 1)} + \frac{1}{2}\mathbf{d}^T{}_{(1\times n)}\mathbf{d}_{(n\times 1)}$$

$$\textit{Subject to} \quad \mathbf{N}^T{}_{(p\times n)}\mathbf{d}_{(n\times 1)} = \mathbf{e}_{(p\times 1)}$$

$$\mathbf{A}^T{}_{(m\times n)}\mathbf{d}_{(n\times 1)} \le \mathbf{b}_{(m\times 1)}$$

- ➡ $\mathbf{H}_{(n\times n)} = \mathbf{I}_{(n\times n)}$이므로 목적 함수는 2차 형식이다.
- ➡ 모든 제약 조건은 선형이다.
- ➡ 이러한 문제는 **볼록 계획**(Convex Programming) **문제**라고 하며, 해가 존재하면 이는 유일한 해 (전역 최적해)이다.

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(1/첫번째 QP문제)

2차 계획 문제의 정의
- 목적 함수: 2차 형식
- 제약 조건: 1차 형식

QP 문제의 풀이를 통한 탐색 방향($d^{(0)}$)의 결정

제약 최적화 문제 (근사화 하기 전)

$$\textit{Minimize} \quad f(\mathbf{x}) = x_1^2 + x_2^2 - 3x_1x_2$$

$$\textit{Subject to} \quad g_1(\mathbf{x}) = \frac{1}{6}x_1^2 + \frac{1}{6}x_2^2 - 1.0 \le 0$$

$$g_2(\mathbf{x}) = -x_1 \le 0$$

$$g_3(\mathbf{x}) = -x_2 \le 0$$

①

$f(1,1) = -1, g_1(1,1) = -\frac{2}{3},$
$g_2(1,1) = -1, g_3(1,1) = -1$
$\nabla f = (-1,-1), \nabla g_1 = (\frac{1}{3},\frac{1}{3}),$
$\nabla g_2 = (-1,0), \nabla g_3 = (0,-1)$

2차 계획 문제

$$\textit{Minimize} \quad \bar{f} = (-d_1 - d_2) + 0.5(d_1^2 + d_2^2)$$

$$\textit{Subject to} \quad \tfrac{1}{3}d_1 + \tfrac{1}{3}d_2 \le \tfrac{2}{3}$$

$$-d_1 \le 1$$

$$-d_2 \le 1$$

여기서,
$d_1 = x_1 - 1, \; d_2 = x_2 - 1$

②

Lagrange 함수

$$L = (-d_1 - d_2) + 0.5(d_1^2 + d_2^2)$$
$$+ u_1[\tfrac{1}{3}(d_1 + d_2 - 2) + s_1^2]$$
$$+ u_2(-d_1 - 1 + s_2^2)$$
$$+ u_3(-d_2 - 1 + s_3^2)$$

③

Kuhn-Tucker 필요 조건:

$$\frac{\partial L}{\partial d_1} = -1 + d_1 + \tfrac{1}{3}u_1 - u_2 = 0$$

$$\frac{\partial L}{\partial d_2} = -1 + d_2 + \tfrac{1}{3}u_1 - u_3 = 0$$

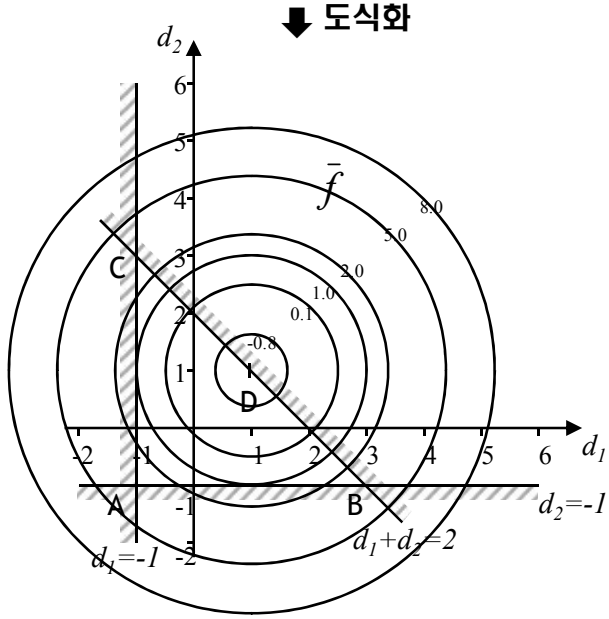
$$\frac{\partial L}{\partial u_1} = \tfrac{1}{3}(d_1 + d_2 - 2) + s_1^2 = 0$$

$$\frac{\partial L}{\partial u_2} = -d_1 - 1 + s_2^2 = 0$$

$$\frac{\partial L}{\partial u_3} = -d_2 - 1 + s_3^2 = 0$$

$$\frac{\partial L}{\partial s_i} = u_i s_i = 0, \; u_i \ge 0, \; i = 1,2,3$$

⬇ 도식화



Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(2/첫번째 QP문제)

2차 계획 문제

$$\textit{Minimize}\ \bar{f} = (-d_1 - d_2) + 0.5(d_1^2 + d_2^2)$$

$$\textit{Subject to}\ \tfrac{1}{3}d_1 + \tfrac{1}{3}d_2 \le \tfrac{2}{3}$$

$$-d_1 \le 1$$

$$-d_2 \le 1$$

Kuhn-Tucker **필요 조건:**

$$\frac{\partial L}{\partial d_1} = -1 + d_1 + \tfrac{1}{3}u_1 - u_2 = 0$$

$$\frac{\partial L}{\partial d_2} = -1 + d_2 + \tfrac{1}{3}u_1 - u_3 = 0$$

$$\frac{\partial L}{\partial u_1} = \tfrac{1}{3}(d_1 + d_2 - 2) + s_1^2 = 0$$

$$\frac{\partial L}{\partial u_2} = -d_1 - 1 + s_2^2 = 0$$

$$\frac{\partial L}{\partial u_3} = -d_2 - 1 + s_3^2 = 0$$

$$\frac{\partial L}{\partial s_i} = u_i s_i = 0,\ u_i \ge 0,\ i = 1,2,3$$

양변에 s_i를 곱한다.

$$\Rightarrow\ u_i s_i^2 = 0,\ u_i \ge 0,\ i = 1,2,3$$

s_i^2을 s_i' 으로 치환

$$s_i^2 = s_i' \ge 0$$

Kuhn-Tucker **필요 조건:**

$$\frac{\partial L}{\partial d_1} = -1 + d_1 + \tfrac{1}{3}u_1 - u_2 = 0$$

$$\frac{\partial L}{\partial d_2} = -1 + d_2 + \tfrac{1}{3}u_1 - u_3 = 0$$

$$\frac{\partial L}{\partial u_1} = \tfrac{1}{3}(d_1 + d_2 - 2) + s_1' = 0$$

$$\frac{\partial L}{\partial u_2} = -d_1 - 1 + s_2' = 0$$

$$\frac{\partial L}{\partial u_3} = -d_2 - 1 + s_3' = 0$$

$$\frac{\partial L}{\partial s_i} = u_i s_i' = 0$$

$$u_i, s_i' \ge 0,\ i = 1,2,3$$

편의상 s_i' 을 s_i 이라고 표현

Kuhn-Tucker **필요 조건:**

$$\frac{\partial L}{\partial d_1} = -1 + d_1 + \tfrac{1}{3}u_1 - u_2 = 0$$

$$\frac{\partial L}{\partial d_2} = -1 + d_2 + \tfrac{1}{3}u_1 - u_3 = 0$$

$$\frac{\partial L}{\partial u_1} = \tfrac{1}{3}(d_1 + d_2 - 2) + s_1 = 0$$

$$\frac{\partial L}{\partial u_2} = -d_1 - 1 + s_2 = 0$$

$$\frac{\partial L}{\partial u_3} = -d_2 - 1 + s_3 = 0$$

$$\frac{\partial L}{\partial s_i} = u_i s_i = 0$$

$$u_i, s_i \ge 0,\ i = 1,2,3$$

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(3/첫번째 QP문제)

2차 계획 문제

$$\textit{Minimize}\quad \bar{f} = (-d_1 - d_2) + 0.5(d_1^2 + d_2^2)$$

$$\textit{Subject to}\quad \tfrac{1}{3}d_1 + \tfrac{1}{3}d_2 \le \tfrac{2}{3}$$

$$-d_1 \le 1$$

$$-d_2 \le 1$$

Matrix 형태로 표현

$$\textit{Minimize}\quad \bar{f} = \mathbf{c}^T{}_{(1\times 2)}\mathbf{d}_{(2\times 1)} + \frac{1}{2}\mathbf{d}^T{}_{(1\times 2)}\mathbf{H}_{(2\times 2)}\mathbf{d}_{(2\times 1)}$$

$$\textit{Subject to}\quad \mathbf{A}^T{}_{(3\times 2)}\mathbf{d}_{(2\times 1)} \le \mathbf{b}_{(3\times 1)}$$

$\mathbf{H}_{(2\times 2)}$ 를 $\mathbf{I}_{(2\times 2)}$ 로 가정한다.

여기서,

$$\mathbf{d}_{(2\times 1)} = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}, \mathbf{c}_{(2\times 1)} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \mathbf{H}_{(2\times 2)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$\mathbf{A}_{(2\times 3)} = \begin{bmatrix} \frac{1}{3} & -1 & 0 \\ \frac{1}{3} & 0 & -1 \end{bmatrix}, \mathbf{b}_{(3\times 1)} = \begin{bmatrix} \frac{2}{3} \\ 1 \\ 1 \end{bmatrix}$$

Kuhn-Tucker 필요 조건:

$$\frac{\partial L}{\partial d_1} = -1 + d_1 + \tfrac{1}{3}u_1 - u_2 = 0$$

$$\frac{\partial L}{\partial d_2} = -1 + d_2 + \tfrac{1}{3}u_1 - u_3 = 0$$

$$\frac{\partial L}{\partial u_1} = \tfrac{1}{3}(d_1 + d_2 - 2) + s_1 = 0$$

$$\frac{\partial L}{\partial u_2} = -d_1 - 1 + s_2 = 0$$

$$\frac{\partial L}{\partial u_3} = -d_2 - 1 + s_3 = 0$$

$$\frac{\partial L}{\partial s_i} = u_i s_i = 0$$

$$u_i, s_i \ge 0,\ i = 1,2,3$$

Kuhn-Tucker **필요조건을 d, c, H, A, b를 이용하여 Matrix 형태로 표현하면?**

Simplex 방법을 이용,탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(4/첫번째 QP문제)

$$\mathbf{d}_{(2\times1)}=\begin{bmatrix}d_1\\d_2\end{bmatrix},\mathbf{c}_{(2\times1)}=\begin{bmatrix}-1\\-1\end{bmatrix},\mathbf{H}_{(2\times2)}=\begin{bmatrix}1&0\\0&1\end{bmatrix},$$

$$\mathbf{A}_{(2\times3)}=\begin{bmatrix}\frac{1}{3}&-1&0\\\frac{1}{3}&0&-1\end{bmatrix},\mathbf{b}_{(3\times1)}=\begin{bmatrix}\frac{2}{3}\\1\\1\end{bmatrix}$$

Kuhn-Tucker **필요 조건:**

$$\frac{\partial L}{\partial d_1}=-1+d_1+\tfrac{1}{3}u_1-u_2=0$$

$$\frac{\partial L}{\partial d_2}=-1+d_2+\tfrac{1}{3}u_1-u_3=0$$

$$\frac{\partial L}{\partial u_1}=\tfrac{1}{3}(d_1+d_2-2)+s_1=0$$

$$\frac{\partial L}{\partial u_2}=-d_1-1+s_2=0$$

$$\frac{\partial L}{\partial u_3}=-d_2-1+s_3=0$$

$$\frac{\partial L}{\partial s_i}=u_i s_i=0$$

$$u_i,s_i\geq0,\ i=1,2,3$$

$$\begin{bmatrix}-1+d_1+\frac{1}{3}u_1-u_2\\-1+d_2+\frac{1}{3}u_1-u_3\end{bmatrix}=\begin{bmatrix}-1\\-1\end{bmatrix}+\begin{bmatrix}1&0\\0&1\end{bmatrix}\begin{bmatrix}d_1\\d_2\end{bmatrix}+\begin{bmatrix}\frac{1}{3}&-1&0\\\frac{1}{3}&0&-1\end{bmatrix}\begin{bmatrix}u_1\\u_2\\u_3\end{bmatrix}$$

$$=\mathbf{c}_{(2\times1)}+\mathbf{H}_{(2\times2)}\mathbf{d}_{(2\times1)}+\mathbf{A}_{(2\times3)}\mathbf{u}_{(3\times1)}=\mathbf{0}$$

$$\begin{bmatrix}\frac{1}{3}(d_1+d_2-2)+s_1\\-d_1-1+s_2\\-d_2-1+s_3\end{bmatrix}=\begin{bmatrix}\frac{1}{3}&\frac{1}{3}\\-1&0\\0&-1\end{bmatrix}\begin{bmatrix}d_1\\d_2\end{bmatrix}+\begin{bmatrix}s_1\\s_2\\s_3\end{bmatrix}-\begin{bmatrix}\frac{2}{3}\\1\\1\end{bmatrix}$$

$$=\mathbf{A}^T{}_{(3\times2)}\mathbf{d}_{(2\times1)}+\mathbf{s}_{(3\times1)}-\mathbf{b}_{(3\times1)}=\mathbf{0}$$

한편, 설계 변수 d는 부호의 제한이 없으므로
Simplex 방법을 이용하기 위해 다음과 같이 표현해야 함

$$\mathbf{d}_{(2\times1)}=\mathbf{d}^{+}_{(2\times1)}-\mathbf{d}^{-}_{(2\times1)}$$

Matrix **형태로 표현**

$$\underbrace{\begin{bmatrix}\mathbf{H}_{(2\times2)}&-\mathbf{H}_{(2\times2)}&\mathbf{A}_{(2\times3)}&\mathbf{0}_{(2\times3)}\\\mathbf{A}^T{}_{(3\times2)}&-\mathbf{A}^T{}_{(3\times2)}&\mathbf{0}_{(3\times3)}&\mathbf{I}_{(3\times3)}\end{bmatrix}}_{=\mathbf{B}_{(5\times10)}}\underbrace{\begin{bmatrix}\mathbf{d}^{+}_{(2\times1)}\\\mathbf{d}^{-}_{(2\times1)}\\\mathbf{u}_{(3\times1)}\\\mathbf{s}_{(3\times1)}\end{bmatrix}}_{=\mathbf{X}_{(10\times1)}}=\underbrace{\begin{bmatrix}-\mathbf{c}_{(2\times1)}\\\mathbf{b}_{(3\times1)}\end{bmatrix}}_{=\mathbf{D}_{(5\times1)}}$$

Simplex 방법을 이용,탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(5/첫번째 QP문제)

Kuhn-Tucker **필요 조건:** $\nabla L(\mathbf{d}^+,\mathbf{d}^-,\mathbf{u},\mathbf{s})=\mathbf{0}$

$$\underbrace{\begin{bmatrix} \mathbf{H}_{(2\times2)} & -\mathbf{H}_{(2\times2)} & \mathbf{A}_{(2\times3)} & \mathbf{0}_{(2\times3)} \\ \mathbf{A}^T{}_{(3\times2)} & -\mathbf{A}^T{}_{(3\times2)} & \mathbf{0}_{(3\times3)} & \mathbf{I}_{(3\times3)} \end{bmatrix}}_{=\mathbf{B}_{(5\times10)}} \underbrace{\begin{bmatrix} \mathbf{d}^+_{(2\times1)} \\ \mathbf{d}^-_{(2\times1)} \\ \mathbf{u}_{(3\times1)} \\ \mathbf{s}_{(3\times1)} \end{bmatrix}}_{=\mathbf{X}_{(10\times1)}} = \underbrace{\begin{bmatrix} -\mathbf{c}_{(2\times1)} \\ \mathbf{b}_{(3\times1)} \end{bmatrix}}_{=\mathbf{D}_{(5\times1)}}$$

여기서, $\mathbf{d}^+_{(2\times1)}=\begin{bmatrix} d_1^+ \\ d_2^+ \end{bmatrix}, \mathbf{d}^-_{(2\times1)}=\begin{bmatrix} d_1^- \\ d_2^- \end{bmatrix}, \mathbf{c}_{(2\times1)}=\begin{bmatrix} -1 \\ -1 \end{bmatrix}, \mathbf{H}_{(2\times2)}=\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{A}_{(2\times3)}=\begin{bmatrix} \frac{1}{3} & -1 & 0 \\ \frac{1}{3} & 0 & -1 \end{bmatrix}, \mathbf{b}_{(3\times1)}=\begin{bmatrix} \frac{2}{3} \\ 1 \\ 1 \end{bmatrix}$

$$\mathbf{B}_{(5\times10)}=\left[\begin{array}{cccc|ccc|ccc} 1 & 0 & -1 & 0 & \frac{1}{3} & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & \frac{1}{3} & 0 & -1 & 0 & 0 & 0 \\ \hline \frac{1}{3} & \frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right]$$

$$\mathbf{X}^T{}_{(1\times10)}=\begin{bmatrix} d_1^+ & d_2^+ & d_1^- & d_2^- & u_1 & u_2 & u_3 & s_1 & s_2 & s_3 \end{bmatrix}, \mathbf{D}^T{}_{(1\times5)}=\begin{bmatrix} 1 & 1 & \frac{2}{3} & 1 & 1 \end{bmatrix}$$

Simplex 방법을 이용,탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(6/첫번째 QP문제)

Kuhn-Tucker 필요 조건(행렬식 표현)

$$\mathbf{B}_{(5\times10)}\mathbf{X}_{(10\times1)} = \mathbf{D}_{(5\times1)}$$

$$\begin{bmatrix} 1 & 0 & -1 & 0 & \frac{1}{3} & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & \frac{1}{3} & 0 & -1 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_1^+ \\ d_2^+ \\ d_1^- \\ d_2^- \\ u_1 \\ u_2 \\ u_3 \\ s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \frac{2}{3} \\ 1 \\ 1 \end{bmatrix}$$

↑ 구하려는 값

➡ **X를 구하는 위 문제는 등호 제약 조건만으로 이루어진 선형 계획 문제임**

➡ $u_i s_i = 0; i = 1\ to\ 3$: **Simplex 방법을 이용하여 부정 선형 연립 방정식에서 구한 해가 이 비선형 부정 방정식을 만족하는지 확인하여 해를 확정한다**

Simplex 방법을 이용,탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(7/첫번째 QP문제)

Simplex 방법을 이용한 QP 문제의 해법

1. Kuhn-Tucker 필요 조건의 해를 구하는 문제는 등호 제약 조건만으로 이루어진 부정 선형 연립 방정식의 해를 구하는 문제(선형 계획 문제)임
2. 부정 선형 연립 방정식의 해를 구하기 위하여 Simplex 방법에서 인위 변수 및 인위 목적 함수를 도입하여 초기 기저 가능해를 구하는 방법임

$$\mathbf{B}_{(5\times10)}\mathbf{X}_{(10\times1)} + \underbrace{\mathbf{Y}_{(5\times1)}}_{\text{인위 변수}} = \mathbf{D}_{(5\times1)}$$

3. 인위 목적 함수는 다음과 같이 정의함

$$w = \sum_{i=1}^{5} Y_i = \sum_{i=1}^{5} D_i - \sum_{j=1}^{10}\sum_{i=1}^{5} B_{ij}X_j = w_0 + \sum_{j=1}^{10} C_j X_j$$

여기서, $C_j = -\sum_{i=1}^{5} B_{ij}$: 행렬 B의 j번째 열의 요소를 모두 더해 부호를 바꾼 것(상대 비용 계수)

$$w_0 = \sum_{i=1}^{5} D_i = 1+1+\frac{2}{3}+1+1 = \frac{14}{3}$$

: 인위 목적 함수의 초기값으로 행렬 D의 모든 요소를 더한 것

4. Simplex 방법을 이용하여 해를 구하고 다음 식을 만족하는지 확인함

$u_i s_i = 0; i = 1\ to\ 3$: Simplex 방법을 이용하여 부정 선형 연립 방정식에서 구한 해가 이 비선형 부정 방정식을 만족하는지 확인하여 해를 확정한다.

Simplex 방법을 이용,탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(8/첫번째 QP문제)

$$\mathbf{B}_{(5\times10)}\mathbf{X}_{(10\times1)} + \underbrace{\mathbf{Y}_{(5\times1)}}_{\text{인위 변수}} = \mathbf{D}_{(5\times1)} \Rightarrow \begin{bmatrix} 1 & 0 & -1 & 0 & \frac{1}{3} & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & \frac{1}{3} & 0 & -1 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_1^+(=X_1) \\ d_2^+(=X_2) \\ d_1^-(=X_3) \\ d_2^-(=X_4) \\ u_1(=X_5) \\ u_2(=X_6) \\ u_3(=X_7) \\ s_1(=X_8) \\ s_2(=X_9) \\ s_3(=X_{10}) \end{bmatrix} + \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \frac{2}{3} \\ 1 \\ 1 \end{bmatrix}$$

인위 목적 함수의 구성

1행부터 5행까지 모두 더함: $\frac{1}{3}X_1+\frac{1}{3}X_2-\frac{1}{3}X_3-\frac{1}{3}X_4+\frac{2}{3}X_5-X_6-X_7+X_8+X_9+X_{10}+\underbrace{Y_1+Y_2+Y_3+Y_4+Y_5}_{w}=\frac{14}{3}$

인위 변수의 합을 w로 치환하고 정리: $-\frac{1}{3}X_1-\frac{1}{3}X_2+\frac{1}{3}X_3+\frac{1}{3}X_4-\frac{2}{3}X_5+X_6+X_7-X_8-X_9-X_{10}=w-\frac{14}{3}$

1

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
Y1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
Y2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
Y3	1/3	1/3	-1/3	-1/3	0	0	0	1	0	0	0	0	1	0	0	2/3	**2/3**
Y4	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1	-
Y5	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	-
A. Obj.	-1/3	-1/3	1/3	1/3	-2/3	1	1	**-1**	-1	-1	0	0	0	0	0	w-14/3	-

인위 목적 함수 식 각 열의 요소를 모두 더해 부호를 바꾼 것(예, 1열: -(1+0+1/3-1+0)=-1/3)

Simplex 방법을 이용,탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(9/첫번째 QP문제)

2

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
Y1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
Y2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
X8	1/3	1/3	-1/3	-1/3	0	0	0	1	0	0	0	0	1	0	0	2/3	-
Y4	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1	1
Y5	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	-
A. Obj.	0	0	0	0	-2/3	1	1	0	**-1**	-1	0	0	1	0	0	w-4	-

3

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
Y1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
Y2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
X8	1/3	1/3	-1/3	-1/3	0	0	0	1	0	0	0	0	1	0	0	2/3	-
X9	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1	-
Y5	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1
A. Obj.	-1	0	1	0	-2/3	1	1	0	0	**-1**	0	0	1	1	0	w-3	-

4

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
Y1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	1
Y2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
X8	1/3	1/3	-1/3	-1/3	0	0	0	1	0	0	0	0	1	0	0	2/3	2
X9	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1	-
X10	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	-
A. Obj.	**-1**	-1	1	1	-2/3	1	1	0	0	0	0	0	1	1	1	w-2	-

Simplex 방법을 이용,탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(10/첫번째 QP문제)

5

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
Y2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	1
X8	0	1/3	0	-1/3	-1/9	1/3	0	1	0	0	-1/3	0	1	0	0	1/3	1
X9	0	0	0	0	1/3	-1	0	0	1	0	1	0	0	1	0	2	-
X10	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	-
A. Obj.	0	-1	0	1	-1/3	0	1	0	0	0	1	0	1	1	1	w-1	-

6

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
X2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
X8	0	0	0	0	-2/9	1/3	1/3	1	0	0	-1/3	-1/3	1	0	0	0	-
X9	0	0	0	0	1/3	-1	0	0	1	0	1	0	0	1	0	2	-
X10	0	0	0	0	1/3	0	-1	0	0	1	0	1	0	0	1	2	-
A. Obj.	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	w-0	-

인위 목적 함수가 0이므로
초기 기저 가능해가 구해졌음

Simplex 방법을 이용,탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(10/첫번째 QP문제)

6

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
X2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
X8	0	0	0	0	-2/9	1/3	1/3	1	0	0	-1/3	-1/3	1	0	0	0	-
X9	0	0	0	0	1/3	-1	0	0	1	0	1	0	0	1	0	2	-
X10	0	0	0	0	1/3	0	-1	0	0	1	0	1	0	0	1	2	-
A. Obj.	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	w-0	-

인위 목적 함수가 0이므로
초기 기저 가능해가 구해졌음

$$\mathbf{X}^T_{(1\times 10)} = \begin{bmatrix} d_1^+ & d_2^+ & d_1^- & d_2^- & u_1 & u_2 & u_3 & s_1 & s_2 & s_3 \end{bmatrix}$$

기저 해를 구해보면

$$X_1 = 1,\quad X_2 = 1,\quad X_8 = 0,\quad X_9 = 2,\quad X_{10} = 2$$

비기저 해는 다음과 같다.

$$X_3 = X_4 = X_5 = X_6 = X_7 = 0$$

한편, 이들은 제약 조건 $X_i X_{3+i} = 0; i = 5\ to\ 7, X_i \geq 0; i = 1\ to\ 10$ 을 모두 만족한다.

따라서 주어진 문제의 최적해는 $d_1 = d_2 = 1, u_1 = u_2 = u_3 = 0, s_1 = 0, s_2 = s_3 = 2$이다.

여기서 u_1과 s_1의 값이 동시에 0인 이유는 무엇인가?

➡ Pivot 과정 중 선택 가능한 열 또는 행 또는 "b_i/a_i"의 계수가 중복인 경우,
어떤 것을 선택하느냐에 따라 다른 초기 기저 가능해가 얻어질 수 있음

➡ 비선형 방정식($u_i * s_i = 0$)을 만족하는 해가 나올 때까지 모든 경우를 확인해 봐야 함

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(11/첫번째 QP문제)

$$\textit{Minimize} \quad f(\mathbf{x}) = x_1^2 + x_2^2 - 3x_1x_2$$
$$\textit{Subject to} \quad g_1(\mathbf{x}) = \frac{1}{6}x_1^2 + \frac{1}{6}x_2^2 - 1.0 \le 0$$
$$g_2(\mathbf{x}) = -x_1 \le 0$$
$$g_3(\mathbf{x}) = -x_2 \le 0$$

주어진 문제의 최적해는 $d_1 = d_2 = 1, u_1 = u_2 = u_3 = 0, s_1 = 0, s_2 = s_3 = 2$ 이다.
여기서 u_1과 s_1의 값이 동시에 0인 이유는 무엇인가?

2차 계획 문제

$$\textit{Minimize} \quad \bar{f} = (-d_1 - d_2) + 0.5(d_1^2 + d_2^2)$$
$$\textit{Subject to} \quad \tfrac{1}{3}d_1 + \tfrac{1}{3}d_2 \le \tfrac{2}{3}$$
$$-d_1 \le 1$$
$$-d_2 \le 1$$

본 예제의 첫번째 QP문제를 도식화 하면 오른쪽과 같다.

$\longrightarrow s_1 = 0$

제약조건 중 $g_1(\mathbf{x})$를 근사화한 $d_1+d_2=2$ 위에 최적점이 존재 한다.

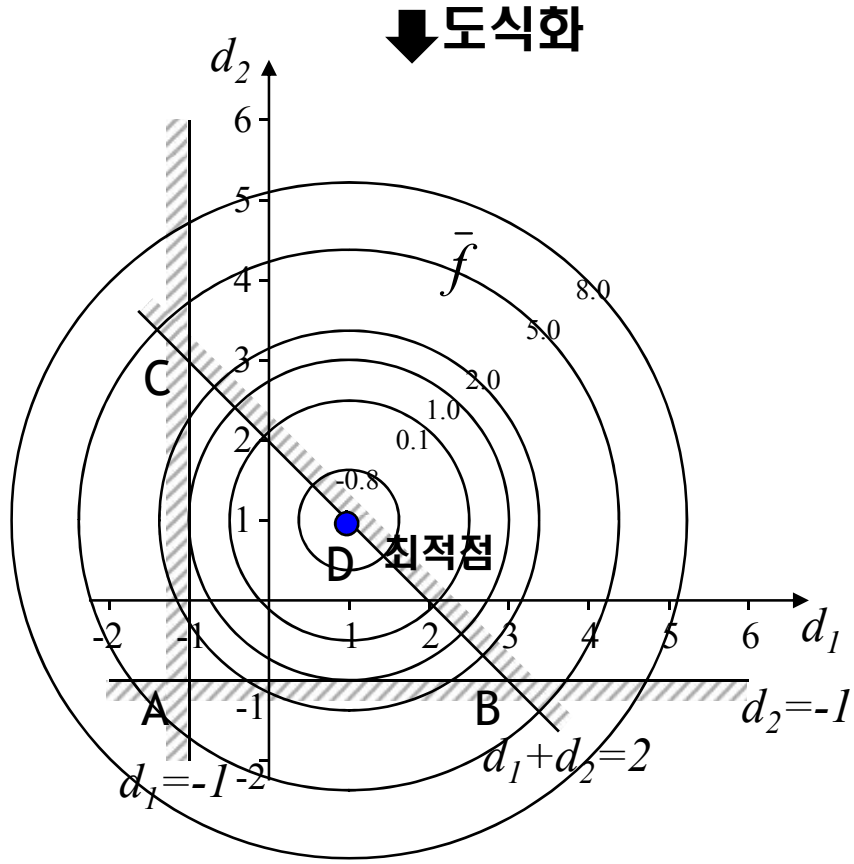
$\longrightarrow u_2 = u_3 = 0$

부등호 제약 조건이 작용하지 않는 가능 영역 안에 최적해가 존재할 것이다.

$\longrightarrow u_1 = 0$

제약조건 $g_1(\mathbf{x})$ 상에 최적점이 존재하나,
최적점의 위치가 2차로 근사화된 목적 함수 최적점의 위치와 동일하다.
따라서 $g_1(\mathbf{x})$이 없는 경우에도 QP문제의 최적점은 변하지 않는다.
($g_1(\mathbf{x})$는 본 문제의 최적점에 영향을 주지 않음)

도식화



Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(1/두번째 QP문제)

2차 계획 문제의 정의
- 목적 함수: 2차 형식
- 제약 조건: 1차 형식

QP 문제의 풀이를 통한 탐색 방향($d^{(1)}$)의 결정

제약 최적화 문제 (근사화 하기 전)

$$\textit{Minimize}\quad f(\mathbf{x}) = x_1^2 + x_2^2 - 3x_1x_2$$

$$\textit{Subject to}\quad g_1(\mathbf{x}) = \frac{1}{6}x_1^2 + \frac{1}{6}x_2^2 - 1.0 \le 0$$

$$g_2(\mathbf{x}) = -x_1 \le 0$$

$$g_3(\mathbf{x}) = -x_1 \le 0$$

①

$$f(1.732,1.732) = -3, \nabla f = (-1.732,-1.732)$$
$$g_1(1.732,1.732) = -5.866\times10^{-5}, \nabla g_1 = (0.577,0.577)$$
$$g_2(1.732,1.732) = -1.732, \nabla g_2 = (-1,0)$$
$$g_3(1.732,1.732) = -1.732, \nabla g_3 = (0,-1)$$

2차 계획 문제

$$\textit{Minimize}\quad \bar{f} = (-1.732d_1 - 1.732d_2) + 0.5(d_1^2 + d_2^2)$$

$$\textit{Subject to}\quad 0.577d_1 + 0.577d_2 \le 5.866\times10^{-5}$$

$$-d_1 \le 1.732$$

$$-d_2 \le 1.732$$

여기서,

$$d_1 = x_1 - 1.732,$$

$$d_2 = x_2 - 1.732$$

②

Lagrange 함수

$$L = (-1.732d_1 - 1.732d_2) + 0.5(d_1^2 + d_2^2) + u_1[0.577(d_1 + d_2) - 5.866\times10^{-5} + s_1^2] + u_2(-d_1 - 1.732 + s_2^2) + u_3(-d_2 - 1.732 + s_3^2)$$

③

Kuhn-Tucker 필요 조건: $\nabla L(\mathbf{d},\mathbf{u},\mathbf{s}) = \mathbf{0}$

$$\frac{\partial L}{\partial d_1} = -1.732 + d_1 + 0.577u_1 - u_2 = 0$$

$$\frac{\partial L}{\partial d_2} = -1.732 + d_2 + 0.577u_1 - u_3 = 0$$

$$\frac{\partial L}{\partial u_1} = 0.577(d_1 + d_2) - 5.866\times10^{-6} + s_1^2 = 0$$

$$\frac{\partial L}{\partial u_2} = -d_1 - 1.732 + s_2^2 = 0$$

$$\frac{\partial L}{\partial u_3} = -d_2 - 1.732 + s_3^2 = 0$$

$$\frac{\partial L}{\partial s_i} = u_i s_i = 0,\ u \ge 0,\ i = 1,2,3$$

1. 양변에 s_i를 곱한 후 s_i^2을 s_i'으로 치환
2. 편의상 s_i'을 s_i로 표기

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(2/두번째 QP문제)

2차 계획 문제의 정의
- 목적 함수: 2차 형식
- 제약 조건: 1차 형식

2차 계획 문제

$$\textit{Minimize} \quad \bar{f} = (-1.732d_1 - 1.732d_2) + 0.5(d_1^2 + d_2^2)$$

$$\textit{Subject to} \quad 0.577d_1 + 0.577d_2 \le 0$$

$$-d_1 \le 1.732$$

$$-d_2 \le 1.732$$

Matrix 형태로 표현

$$\textit{Minimize} \quad \bar{f} = \mathbf{c}^T_{(1\times2)}\mathbf{d}_{(2\times1)} + \frac{1}{2}\mathbf{d}^T_{(1\times2)}\mathbf{H}_{(2\times2)}\mathbf{d}_{(2\times1)}$$

$$\textit{Subject to} \quad \mathbf{A}^T_{(3\times2)}\mathbf{d}_{(2\times1)} \le \mathbf{b}_{(3\times1)}$$

Kuhn-Tucker 필요 조건:

$$\frac{\partial L}{\partial d_1} = -1.732 + d_1 + 0.577u_1 - u_2 = 0$$

$$\frac{\partial L}{\partial d_2} = -1.732 + d_2 + 0.577u_1 - u_3 = 0$$

$$\frac{\partial L}{\partial u_1} = 0.577(d_1 + d_2) - 5.866\times10^{-6} + s_1 = 0$$

$$\frac{\partial L}{\partial u_2} = -d_1 - 1.732 + s_2 = 0$$

$$\frac{\partial L}{\partial u_3} = -d_2 - 1.732 + s_3 = 0$$

$$\frac{\partial L}{\partial s_i} = u_i s_i = 0,\ u_i, s_i \ge 0,\ i = 1,2,3$$

Kuhn-Tucker 필요조건을 d, c, H, A, b를 이용하여 Matrix 형태로 표현하면?

$\mathbf{H}_{(2\times2)}$ 를 $\mathbf{I}_{(2\times2)}$ 로 가정한다.

여기서,

$$\mathbf{d}_{(2\times1)} = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}, \mathbf{c}_{(2\times1)} = \begin{bmatrix} -1.732 \\ -1.732 \end{bmatrix}, \mathbf{H}_{(2\times2)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$\mathbf{A}_{(2\times3)} = \begin{bmatrix} 0.577 & -1 & 0 \\ 0.577 & 0 & -1 \end{bmatrix}, \mathbf{b}_{(3\times1)} = \begin{bmatrix} 0 \\ 1.732 \\ 1.732 \end{bmatrix}$$

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(3/두번째 QP문제)

$$\mathbf{d}_{(2\times1)}=\begin{bmatrix}d_1\\d_2\end{bmatrix},\ \mathbf{c}_{(2\times1)}=\begin{bmatrix}-1.732\\-1.732\end{bmatrix},\ \mathbf{H}_{(2\times2)}=\begin{bmatrix}1&0\\0&1\end{bmatrix},$$

$$\mathbf{A}_{(2\times3)}=\begin{bmatrix}0.577&-1&0\\0.577&0&-1\end{bmatrix},\ \mathbf{b}_{(3\times1)}=\begin{bmatrix}0\\1.732\\1.732\end{bmatrix}$$

Kuhn-Tucker 필요 조건:

$$\frac{\partial L}{\partial d_1}=-1.732+d_1+0.577u_1-u_2=0$$

$$\frac{\partial L}{\partial d_2}=-1.732+d_2+0.577u_1-u_3=0$$

$$\frac{\partial L}{\partial u_1}=0.577\left(d_1+d_2\right)-5.866\times10^{-6}+s_1=0$$

$$\frac{\partial L}{\partial u_2}=-d_1-1.732+s_2=0$$

$$\frac{\partial L}{\partial u_3}=-d_2-1.732+s_3=0$$

$$\frac{\partial L}{\partial s_i}=u_i s_i=0,\ u_i,s_i\geq0,\ i=1,2,3$$

$$\begin{bmatrix}-1.732+d_1+0.577u_1-u_2\\-1.732+d_2+0.577u_1-u_3\end{bmatrix}$$

$$=\begin{bmatrix}-1.732\\-1.732\end{bmatrix}+\begin{bmatrix}1&0\\0&1\end{bmatrix}\begin{bmatrix}d_1\\d_2\end{bmatrix}+\begin{bmatrix}0.577&-1&0\\0.577&0&-1\end{bmatrix}\begin{bmatrix}u_1\\u_2\\u_3\end{bmatrix}$$

$$=\mathbf{c}_{(2\times1)}+\mathbf{H}_{(2\times2)}\mathbf{d}_{(2\times1)}+\mathbf{A}_{(2\times3)}\mathbf{u}_{(3\times1)}=\mathbf{0}$$

$$\begin{bmatrix}-0.577d_1-0.577d_2+s_1\\-d_1-1.732+s_2\\-d_2-1.732+s_3\end{bmatrix}=\begin{bmatrix}0.577&0.577\\-1&0\\0&-1\end{bmatrix}\begin{bmatrix}d_1\\d_2\end{bmatrix}+\begin{bmatrix}s_1\\s_2\\s_3\end{bmatrix}-\begin{bmatrix}0\\1.732\\1.732\end{bmatrix}$$

$$=\mathbf{A}^T{}_{(3\times2)}\mathbf{d}_{(2\times1)}+\mathbf{s}_{(3\times1)}-\mathbf{b}_{(3\times1)}=\mathbf{0}$$

한편, 설계 변수 d는 부호의 제한이 없으므로
Simplex 방법을 이용하기 위해 다음과 같이 표현해야 함

$$\mathbf{d}_{(2\times1)}=\mathbf{d}^{+}_{(2\times1)}-\mathbf{d}^{-}_{(2\times1)}$$

Matrix 형태로 표현

$$\underbrace{\begin{bmatrix}\mathbf{H}_{(2\times2)}&-\mathbf{H}_{(2\times2)}&\mathbf{A}_{(2\times3)}&\mathbf{0}_{(2\times3)}\\\mathbf{A}^T{}_{(3\times2)}&-\mathbf{A}^T{}_{(3\times2)}&\mathbf{0}_{(3\times3)}&\mathbf{I}_{(3\times3)}\end{bmatrix}}_{=\mathbf{B}_{(5\times10)}}\underbrace{\begin{bmatrix}\mathbf{d}^{+}_{(2\times1)}\\\mathbf{d}^{-}_{(2\times1)}\\\mathbf{u}_{(3\times1)}\\\mathbf{s}_{(3\times1)}\end{bmatrix}}_{=\mathbf{X}_{(10\times1)}}=\underbrace{\begin{bmatrix}-\mathbf{c}_{(2\times1)}\\\mathbf{b}_{(3\times1)}\end{bmatrix}}_{=\mathbf{D}_{(5\times1)}}$$

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(4/두번째 QP문제)

Kuhn-Tucker **필요 조건:** $\nabla L(\mathbf{d}^+, \mathbf{d}^-, \mathbf{u}, \mathbf{s}) = \mathbf{0}$

$$\underbrace{\begin{bmatrix} \mathbf{H}_{(2\times2)} & -\mathbf{H}_{(2\times2)} & \mathbf{A}_{(2\times3)} & \mathbf{0}_{(2\times3)} \\ \mathbf{A}^T{}_{(3\times2)} & -\mathbf{A}^T{}_{(3\times2)} & \mathbf{0}_{(3\times3)} & \mathbf{I}_{(3\times3)} \end{bmatrix}}_{=\mathbf{B}_{(5\times10)}} \underbrace{\begin{bmatrix} \mathbf{d}^+_{(2\times1)} \\ \mathbf{d}^-_{(2\times1)} \\ \mathbf{u}_{(3\times1)} \\ \mathbf{s}_{(3\times1)} \end{bmatrix}}_{=\mathbf{X}_{(10\times1)}} = \underbrace{\begin{bmatrix} -\mathbf{c}_{(2\times1)} \\ \mathbf{b}_{(3\times1)} \end{bmatrix}}_{=\mathbf{D}_{(5\times1)}}$$

여기서, $\mathbf{d}^+_{(2\times1)} = \begin{bmatrix} d_1^+ \\ d_2^+ \end{bmatrix}, \mathbf{d}^-_{(2\times1)} = \begin{bmatrix} d_1^- \\ d_2^- \end{bmatrix}, \mathbf{c}_{(2\times1)} = \begin{bmatrix} -1.732 \\ -1.732 \end{bmatrix}, \mathbf{H}_{(2\times2)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{A}_{(2\times3)} = \begin{bmatrix} 0.577 & -1 & 0 \\ 0.577 & 0 & -1 \end{bmatrix}, \mathbf{b}_{(3\times1)} = \begin{bmatrix} 0 \\ 1.732 \\ 1.732 \end{bmatrix}$

$$\mathbf{B}_{(5\times10)} = \left[\begin{array}{cccc|ccc|ccc} 1 & 0 & -1 & 0 & 0.577 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0.577 & 0 & -1 & 0 & 0 & 0 \\ \hline 0.577 & 0.577 & -0.577 & -0.577 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right]$$

$$\mathbf{X}^T_{(1\times10)} = \begin{bmatrix} d_1^+ & d_2^+ & d_1^- & d_2^- & u_1 & u_2 & u_3 & s_1 & s_2 & s_3 \end{bmatrix}, \mathbf{D}^T_{(1\times5)} = \begin{bmatrix} 1.732 & 1.732 & 0 & 1.732 & 1.732 \end{bmatrix}$$

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(5/두번째 QP문제)

Kuhn-Tucker 필요 조건(행렬식 표현)

$$\mathbf{B}_{(5\times10)}\mathbf{X}_{(10\times1)} = \mathbf{D}_{(5\times1)}$$

$$\begin{bmatrix} 1 & 0 & -1 & 0 & 0.577 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0.577 & 0 & -1 & 0 & 0 & 0 \\ 0.577 & 0.577 & -0.577 & -0.577 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_1^+ \\ d_2^+ \\ d_1^- \\ d_2^- \\ u_1 \\ u_2 \\ u_3 \\ s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} 1.732 \\ 1.732 \\ 0 \\ 1.732 \\ 1.732 \end{bmatrix}$$

구하려는 값

➡ X를 구하는 위 문제는 등호 제약 조건만으로 이루어진 선형 계획 문제임

➡ $u_i s_i = 0; i = 1 \; to \; 3$: Simplex 방법을 이용하여 부정 선형 연립 방정식에서 구한 해가 이 비선형 부정 방정식을 만족하는지 확인하여 해를 확정한다

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(6/두번째 QP문제)

Simplex 방법을 이용한 QP 문제의 해법

1. Kuhn-Tucker 필요 조건의 해를 구하는 문제는 등호 제약 조건만으로 이루어진 부정 선형 연립 방정식의 해를 구하는 문제(선형 계획 문제)임
2. 부정 선형 연립 방정식의 해를 구하기 위하여 Simplex 방법에서 인위 변수 및 인위 목적 함수를 도입하여 초기 기저 가능해를 구하는 방법임

$$\mathbf{B}_{(5\times10)}\mathbf{X}_{(10\times1)} + \underbrace{\mathbf{Y}_{(5\times1)}}_{\text{인위 변수}} = \mathbf{D}_{(5\times1)}$$

3. 인위 목적 함수는 다음과 같이 정의함

$$w = \sum_{i=1}^{5} Y_i = \sum_{i=1}^{5} D_i - \sum_{j=1}^{10}\sum_{i=1}^{5} B_{ij}X_j = w_0 + \sum_{j=1}^{10} C_j X_j$$

여기서, $C_j = -\sum_{i=1}^{5} B_{ij}$: 행렬 B의 j번째 열의 요소를 모두 더해 부호를 바꾼 것(상대 비용 계수)

$$w_0 = \sum_{i=1}^{5} D_i = 1+1+\frac{2}{3}+1+1 = \frac{14}{3}$$

: 인위 목적 함수의 초기값으로 행렬 D의 모든 요소를 더한 것

4. Simplex 방법을 이용하여 해를 구하고 다음 식을 만족하는지 확인함

$u_i s_i = 0; i = 1\ to\ 3$: Simplex 방법을 이용하여 부정 선형 연립 방정식에서 구한 해가 이 비선형 부정 방정식을 만족하는지 확인하여 해를 확정한다.

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(7/두번째 QP문제)

$$\mathbf{B}_{(5\times10)}\mathbf{X}_{(10\times1)} + \underbrace{\mathbf{Y}_{(5\times1)}}_{\text{인위 변수}} = \mathbf{D}_{(5\times1)} \Rightarrow \begin{bmatrix} 1 & 0 & -1 & 0 & 0.577 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0.577 & 0 & -1 & 0 & 0 & 0 \\ 0.577 & 0.577 & -0.577 & -0.577 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_1^+ \\ d_2^+ \\ d_1^- \\ d_2^- \\ u_1 \\ u_2 \\ u_3 \\ s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} 1.732 \\ 1.732 \\ 0 \\ 1.732 \\ 1.732 \end{bmatrix}$$

인위 목적 함수의 구성

1행부터 5행까지 모두 더함: $0.577X_1 + 0.577X_2 - 0.577X_3 - 0.577X_4 + 1.154X_5 - X_6 - X_7 + X_8 + X_9 + X_{10} + \underbrace{Y_1 + Y_2 + Y_3 + Y_4 + Y_5}_{w} = 6.928$

인위 변수의 합을 w로 치환하고 정리: $-0.577X_1 - 0.577X_2 + 0.577X_3 + 0.577X_4 - 1.154X_5 + X_6 + X_7 - X_8 - X_9 - X_{10} = w - 6.928$

1

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
Y1	1	0	-1	0	0.577	-1	0	0	0	0	1	0	0	0	0	1.732	3
Y2	0	1	0	-1	0.577	0	-1	0	0	0	0	1	0	0	0	1.732	3
Y3	0.577	0.577	-0.577	-0.577	0	0	0	**1**	0	0	0	0	1	0	0	0	-
Y4	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1.732	-
Y5	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1.732	-
A. Obj.	-0.577	-0.577	0.577	0.577	-1.154	1	1	**-1**	-1	-1	0	0	0	0	0	w-6.928	-

인위 목적 함수 식 각 열의 요소를 모두 더해 부호를 바꾼 것(예, 1열: -(1+0+1/3-1+0)=-1/3)

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(8/두번째 QP문제)

2

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X5	1.732	0.000	-1.732	0.000	1.000	-1.732	0.000	0.000	0.000	0.000	1.732	0.000	0.000	0.000	0.000	3.000	-1.732
Y2	-1.000	1.000	1.000	-1.000	0.000	1.000	-1.000	0.000	0.000	0.000	-1.000	1.000	0.000	0.000	0.000	0.000	0.000
Y3	0.577	0.577	-0.577	-0.577	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
Y4	-1.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	1.000	0.000	1.732	1.732
Y5	0.000	-1.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	1.000	1.732	-
A. Obj.	1.423	-0.577	-1.423	0.577	0.000	-1.000	1.000	-1.000	-1.000	-1.000	2.000	0.000	0.000	0.000	0.000	w-3.464	

3

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X5	0.000	1.732	0.000	-1.732	1.000	0.000	-1.732	0.000	0.000	0.000	0.000	1.732	0.000	0.000	0.000	3.000	-
X3	-1.000	1.000	1.000	-1.000	0.000	1.000	-1.000	0.000	0.000	0.000	-1.000	1.000	0.000	0.000	0.000	0.000	-
Y3	0.000	1.155	0.000	-1.155	0.000	0.577	-0.577	1.000	0.000	0.000	-0.577	0.577	1.000	0.000	0.000	0.000	-
Y4	0.000	-1.000	0.000	1.000	0.000	-1.000	1.000	0.000	1.000	0.000	1.000	-1.000	0.000	1.000	0.000	1.732	-
Y5	0.000	-1.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	1.000	1.732	1.732
A. Obj.	0.000	0.845	0.000	-0.845	0.000	0.423	-0.423	-1.000	-1.000	-1.000	0.577	1.423	0.000	0.000	0.000	w-3.464	

4

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X5	0.000	1.732	0.000	-1.732	1.000	0.000	-1.732	0.000	0.000	0.000	0.000	1.732	0.000	0.000	0.000	3.000	-
X3	-1.000	1.000	1.000	-1.000	0.000	1.000	-1.000	0.000	0.000	0.000	-1.000	1.000	0.000	0.000	0.000	0.000	-
Y3	0.000	1.155	0.000	-1.155	0.000	0.577	-0.577	1.000	0.000	0.000	-0.577	0.577	1.000	0.000	0.000	0.000	-
Y4	0.000	-1.000	0.000	1.000	0.000	-1.000	1.000	0.000	1.000	0.000	1.000	-1.000	0.000	1.000	0.000	1.732	1.732
X10	0.000	-1.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	1.000	1.732	-
A. Obj.	0.000	-0.155	0.000	0.155	0.000	0.423	-0.423	-1.000	-1.000	0.000	0.577	1.423	0.000	0.000	1.000	w-1.732	

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(9/두번째 QP문제)

5

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X5	0.000	1.732	0.000	-1.732	1.000	0.000	-1.732	0.000	0.000	0.000	0.000	1.732	0.000	0.000	0.000	3.000	1.732
X3	-1.000	1.000	1.000	-1.000	0.000	1.000	-1.000	0.000	0.000	0.000	-1.000	1.000	0.000	0.000	0.000	0.000	0.000
Y3	0.000	1.155	0.000	-1.155	0.000	0.577	-0.577	1.000	0.000	0.000	-0.577	0.577	1.000	0.000	0.000	0.000	0.000
X9	0.000	-1.000	0.000	1.000	0.000	-1.000	1.000	0.000	1.000	0.000	1.000	-1.000	0.000	1.000	0.000	1.732	-1.732
X10	0.000	-1.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	1.000	1.732	-1.732
A. Obj.	0.000	-1.155	0.000	1.155	0.000	-0.577	0.577	-1.000	0.000	0.000	1.577	0.423	0.000	1.000	1.000	w-0.000	

6

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X5	1.732	0.000	-1.732	0.000	1.000	-1.732	0.000	0.000	0.000	0.000	1.732	0.000	0.000	0.000	0.000	3.000	1.732
X2	-1.000	1.000	1.000	-1.000	0.000	1.000	-1.000	0.000	0.000	0.000	-1.000	1.000	0.000	0.000	0.000	0.000	0.000
Y3	1.155	0.000	-1.155	0.000	0.000	-0.577	0.577	1.000	0.000	0.000	0.577	-0.577	1.000	0.000	0.000	0.000	0.000
X9	-1.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	1.000	0.000	1.732	-1.732
X10	-1.000	0.000	1.000	0.000	0.000	1.000	-1.000	0.000	0.000	1.000	-1.000	1.000	0.000	0.000	1.000	1.732	-1.732
A. Obj.	-1.155	0.000	1.155	0.000	0.000	0.577	-0.577	-1.000	0.000	0.000	0.423	1.577	0.000	1.000	1.000	w-0.000	

7

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X5	0.000	0.000	0.000	0.000	1.000	-0.866	-0.866	-1.500	0.000	0.000	0.866	0.866	-1.500	0.000	0.000	3.000	
X2	0.000	1.000	0.000	-1.000	0.000	0.500	-0.500	0.866	0.000	0.000	-0.500	0.500	0.866	0.000	0.000	0.000	
X1	1.000	0.000	-1.000	0.000	0.000	-0.500	0.500	0.866	0.000	0.000	0.500	-0.500	0.866	0.000	0.000	0.000	
X9	0.000	0.000	0.000	0.000	0.000	-0.500	0.500	0.866	1.000	0.000	0.500	-0.500	0.866	1.000	0.000	1.732	
X10	0.000	0.000	0.000	0.000	0.000	0.500	-0.500	0.866	0.000	1.000	-0.500	0.500	0.866	0.000	1.000	1.732	
A. Obj.	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	1.000	1.000	1.000	1.000	w-0.000	

Simplex 방법을 이용, 탐색 방향을 결정하기 위한 2차 계획 문제의 풀이(10/두번째 QP문제)

7

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
X5	0.000	0.000	0.000	0.000	1.000	-0.866	-0.866	-1.500	0.000	0.000	0.866	0.866	-1.500	0.000	0.000	3.000	
X2	0.000	1.000	0.000	-1.000	0.000	0.500	-0.500	0.866	0.000	0.000	-0.500	0.500	0.866	0.000	0.000	0.000	
X1	1.000	0.000	-1.000	0.000	0.000	-0.500	0.500	0.866	0.000	0.000	0.500	-0.500	0.866	0.000	0.000	0.000	
X9	0.000	0.000	0.000	0.000	0.000	-0.500	0.500	0.866	1.000	0.000	0.500	-0.500	0.866	1.000	0.000	1.732	
X10	0.000	0.000	0.000	0.000	0.000	0.500	-0.500	0.866	0.000	1.000	-0.500	0.500	0.866	0.000	1.000	1.732	
A. Obj.	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	1.000	1.000	1.000	1.000	w-0.000	

$$\mathbf{X}^T_{(1\times 10)} = \begin{bmatrix} d_1^+ & d_2^+ & d_1^- & d_2^- & u_1 & u_2 & u_3 & s_1 & s_2 & s_3 \end{bmatrix}$$

기저 해를 구해보면

$$X_5 = 3, \quad X_2 = 0, \quad X_1 = 0, \quad X_9 = 1.732, \quad X_{10} = 1.732$$

비기저 해는 다음과 같다.

$$X_3 = X_4 = X_6 = X_7 = X_8 = 0$$

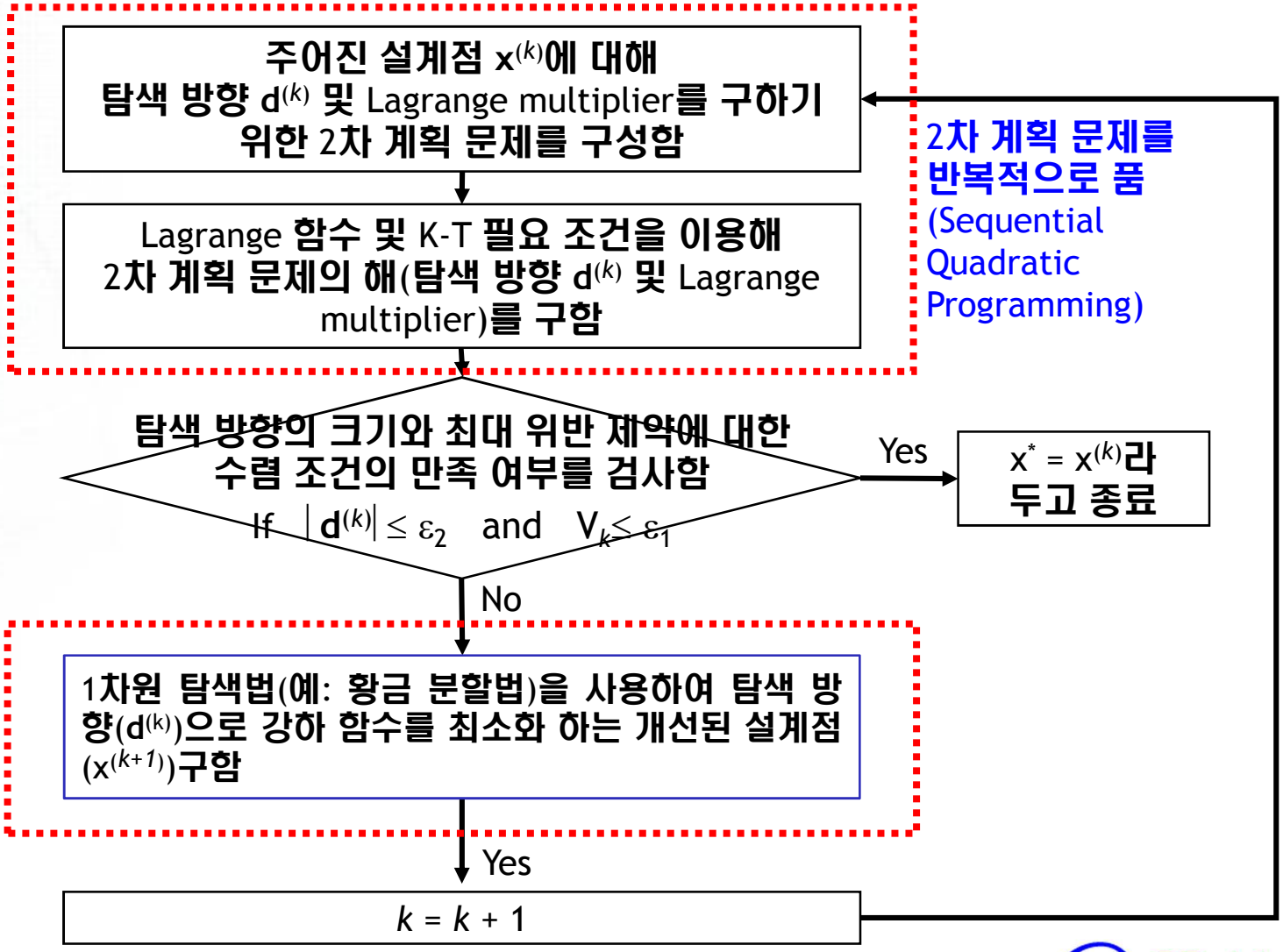
한편, 이들은 제약 조건 $X_i X_{3+i} = 0; i = 5\ to\ 7, X_i \geq 0; i = 1\ to\ 10$ **을 모두 만족한다.**

따라서 주어진 문제의 최적해는 $d_1 = d_2 = 0,\ u_1 = 3,\ u_2 = u_3 = 0,\ s_1 = 0,\ s_2 = s_3 = 1.732$ **이며,**

➡ Pivot 과정 중 선택 가능한 열 또는 행 또는 "b_i/a_i"의 계수가 중복인 경우, 어떤 것을 선택하느냐에 따라 다른 초기 기저 가능해가 얻어질 수 있음

➡ 비선형 방정식($u_i * s_i = 0$)을 만족하는 해가 나올 때까지 모든 경우를 확인해 봐야 함

CSD 알고리즘의 Flow Diagram



CSD 알고리즘의 요약

- 단계 1: k=0으로 둔다. $\mathbf{x}^{(0)}$으로 설계 변수의 초기값을 추정한다. 벌칙 매개 변수 R_0, 허용되는 제약 조건의 위배 정도와 수렴 기준으로 작은 수 ε_1, ε_2의 적절한 초기값을 선정한다.

- 단계 2: $\mathbf{x}^{(k)}$에서 목적 함수, 제약 조건과 이들의 경사도(gradient)를 계산한다. 또한 최대 위배 제약 조건 V_k를 계산한다.

- 단계 3: 목적 함수, 제약 조건과 이들이 경사도를 이용하여 2차 계획 문제를 정의하고, 이를 풀어 탐색 방향 $\mathbf{d}^{(k)}$(= $\mathbf{x}^{(k+1)}$ - $\mathbf{x}^{(k)}$)와 Lagrange multiplier $\mathbf{v}^{(k)}$, $\mathbf{u}^{(k)}$를 구한다.

- 단계 4: 수렴 기준 $|\mathbf{d}^{(k)}| \le \varepsilon_2$을 만족하는지 확인한다. 그리고 최대 위반 제약 조건 $V_k \le \varepsilon_1$을 확인한다. 만일 수렴 기준을 만족하면 현재의 $\mathbf{x}^{(k)}$가 최적해라고 가정하고 종료한다. 그렇지 않다면 다음 단계로 간다.

- 단계 5: Lagrange multiplier의 합 r_k를 계산하여 $R = \max\{R_k, r_k\}$로 둔다.

- 단계 6: 새로운 설계 변수 $\mathbf{x}^{(k,j)} = \mathbf{x}^{(k)} + \alpha_{(k,j)}\mathbf{d}^{(k)}$로 둔다. 여기서 $\alpha = \alpha_{(k,j)}$는 적절한 이동 거리이다. 이동 거리는 탐색 방향 $\mathbf{d}^{(k)}$를 따라 제약 조건의 위배량을 원래 목적 함수에 더한 수정된 목적 함수(강하 함수, descent function)를 최소화 하여 구한다. 1차원 탐색 방법을 이동 거리를 결정하는 데 이용할 수 있다.
  (1차원 탐색이 끝나면 최종 결정된 $\mathbf{x}^{(k,j)}$ 를 $\mathbf{x}^{(k,+1)}$ 로 변경한다.)

- 단계 7: 현재의 벌칙 매개 변수를 $R_k = R$로 저장한다. 반복 횟수를 $k = k+1$로 수정하고 단계 2로 간다.

제약 최적화 문제의 해결을 위한 SQP-CSD Class의 구현 예

- Part 1: Simplex 방법
  - Phase I/Phase II
- Part 2: QP(Quadratic Programming)
  - 주어진 문제의 목적 함수식을 2차 형식으로, 제약 조건식들을 1차 형식으로 근사화 하여 탐색 방향을 구함
  - Kuhn-Tucker 필요 조건을 이용하여, 선형 및 비선형 방정식을 구성하여 이를 풀어 탐색 방향을 구함
    - 선형 부정 방정식: Simplex 방법을 이용하여 해를 구함
    - 비선형 부정 방정식: 선형 부정 방정식으로부터 구한 해가 이 비선형 부정 방정식을 만족하는지 확인하여 해를 확정한다.
- Part 3: SQP-CSD 방법
  - SQP(Sequential Quadratic Programming): QP를 연속적(Sequential)으로 풀어 현재의 설계점에서의 탐색 방향을 구함
  - CSD(Constrained Steepest Descent): 원래의 목적 함수식과 제약 조건식들로부터 강하 함수를 구성하고 이를 최소화 하는 이동 거리를 구함(1차원 탐색 방법을 사용, 예: 황금분할법)

Term Project #8
SQP(Sequential Quadratic Programming)-CSD(Constrained Steepest Descent) Programming Guide

Advanced
Ship
Design
Automation
Laboratory

CSD Programming 과제

- 다음의 비선형 최적화 문제 예시 1~5에 대해 CSD 방법으로 최적해를 구하는 프로그램을 작성한다.
  - 비선형 최적화 문제 예시 1~5: 3~7쪽 참고

- 채점 기준
  - 2차 계획 문제를 이용한 Simplex Table 구성: 30점
  - Simplex 방법을 이용한 탐색 방향 결정: 20점
  - 강하함수와 황금분할법을 이용한 탐색 거리 결정: 20점
  - 각 예제 별로 최적해 계산
    - 예제 1, 2, 3, 4: 각 5점 (총 20점)
    - 예제 5: 10점
  - 감점 항목
    - 프로그램 실행 시 5개의 예제를 선택할 수 있어야 함 (-5점)
    - 매 CSD Algorithm 수행 시 수행 횟수와 x, 그리고 그 때의 목적 함수 값을 출력해야 함 (-10점)
    - Copy 시 0점

CSD Program 실행 결과

프로그램 실행 화면

```
CSD Program
====================================================
(1) f = x1^2 + x2^2 -3*x1*x2
(2) f = x1 - x2 + 2*x1^2 + 2*x1*x2 + x2^2
(3) f = -[ 25- (x1-5)^2 - (x2 -5)^2 ]
(4) f = x1^2 + 2*x2^2 - 4*x1 - 2*x1*x2 + 10
(5) f = golden price
====================================================
문제를 선택하세요(1-6, 끝내려면 0):
```

2번 예제 실행 결과

```
====================================================
Iteration No.:  23
Design Point:      -0.99999912      1.50000059
Objective Function Value:      -1.25000000
====================================================

====================================================
Iteration No.:  24
Design Point:      -1.00000065      1.49999964
Objective Function Value:      -1.25000000
====================================================

====================================================
Iteration No.:  25
Design Point:      -0.99999956      1.50000030
Objective Function Value:      -1.25000000
====================================================

====================================================
Iteration No.:  26
Design Point:      -1.00000035      1.49999979
Objective Function Value:      -1.25000000
====================================================
```

비선형 최적화 프로그램을 이용한 최적 설계 예 (1)

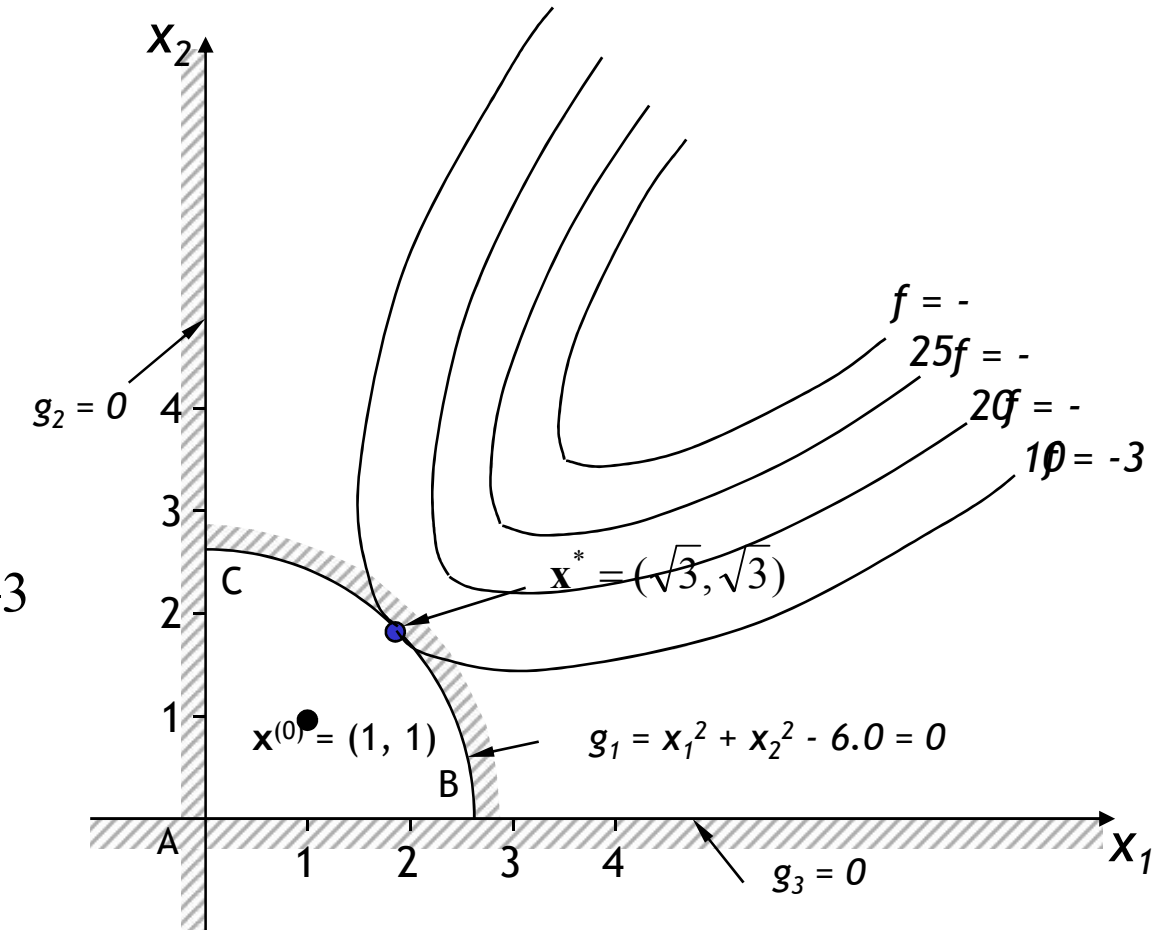
Minimize $f(\mathbf{x}) = x_1^2 + x_2^2 - 3x_1x_2$

Subject to $g_1(\mathbf{x}) = \frac{1}{6}x_1^2 + \frac{1}{6}x_2^2 - 1.0 \le 0$

$g_2(\mathbf{x}) = -x_1 \le 0$

$g_3(\mathbf{x}) = -x_1 \le 0$

최적해는 $\mathbf{x}^* = (\sqrt{3}, \sqrt{3}), f(\mathbf{x}^*) = -3$



비선형 최적화 프로그램을 이용한 최적 설계 예 (2)

Minimize

$$f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$$

Solution

$$x_1^* = -1.0, x_2^* = 1.5, f(x_1^*, x_2^*) = -1.25$$

비선형 최적화 프로그램을 이용한 최적 설계 예 (3)

Minimize

$$f(x_1, x_2) = -\left[25 - (x_1 - 5)^2 - (x_2 - 5)^2\right]$$

Subject to

$$g_1(x_1, x_2) = -32 + 4x_1 + {x_2}^2 \le 0$$
$$g_2(x_1, x_2) = -x_1 \le 0$$
$$g_3(x_1, x_2) = x_1 \le 10$$
$$g_4(x_1, x_2) = -x_2 \le 0$$
$$g_5(x_1, x_2) = x_2 \le 10$$

Solution

$$x_1^* = 4.374, x_2^* = 3.808, f(x_1^*, x_2^*) = -4.815$$

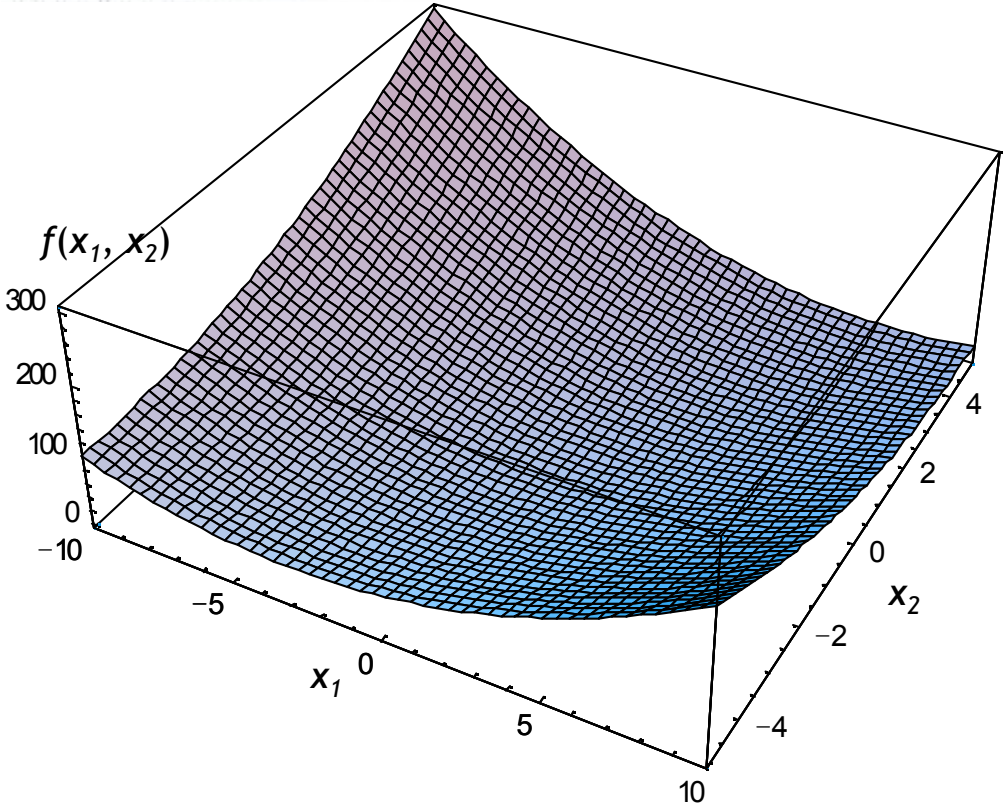
비선형 최적화 프로그램을 이용한 최적 설계 예 (4)

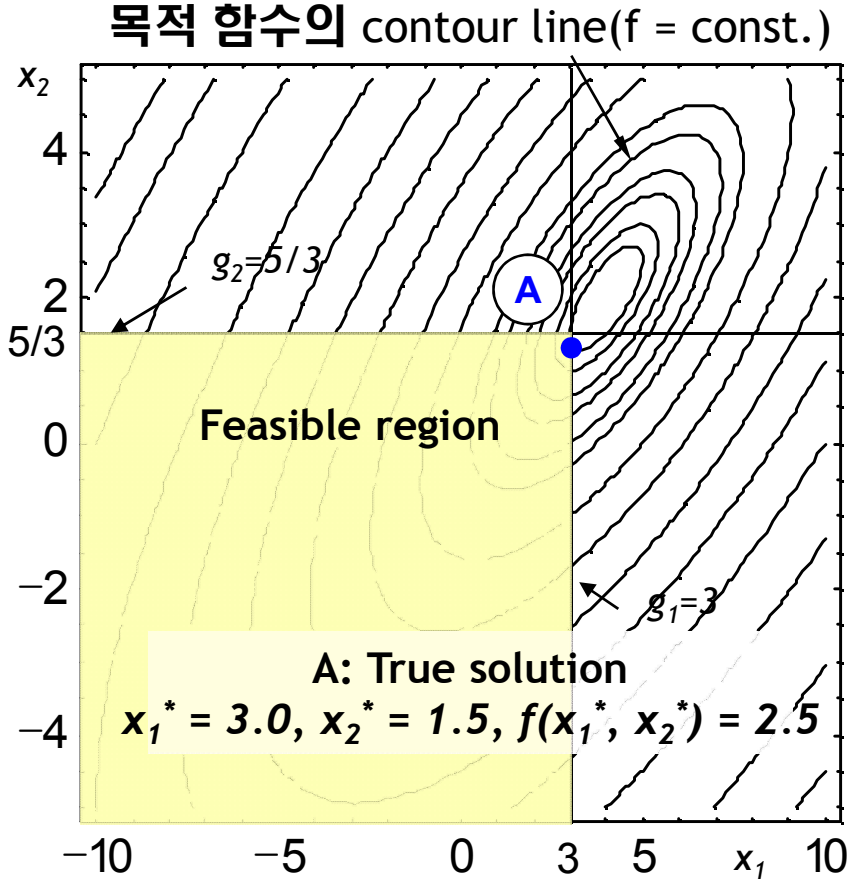
Find $x_1(=B/T), x_2(=1/C_B)$

Minimize $f(x_1, x_2) = x_1^2 + 2x_2^2 - 4x_1 - 2x_1x_2 + 10$

Subject to $g_1(x_1, x_2) = x_1 - 3 \le 0$ ➡ 미지수 2개, 부등호 제약 조건 2개인 최적화 문제

$g_2(x_1, x_2) = x_2 - 5/3 \le 0$





비선형 최적화 프로그램을 이용한 최적 설계 예 (5)

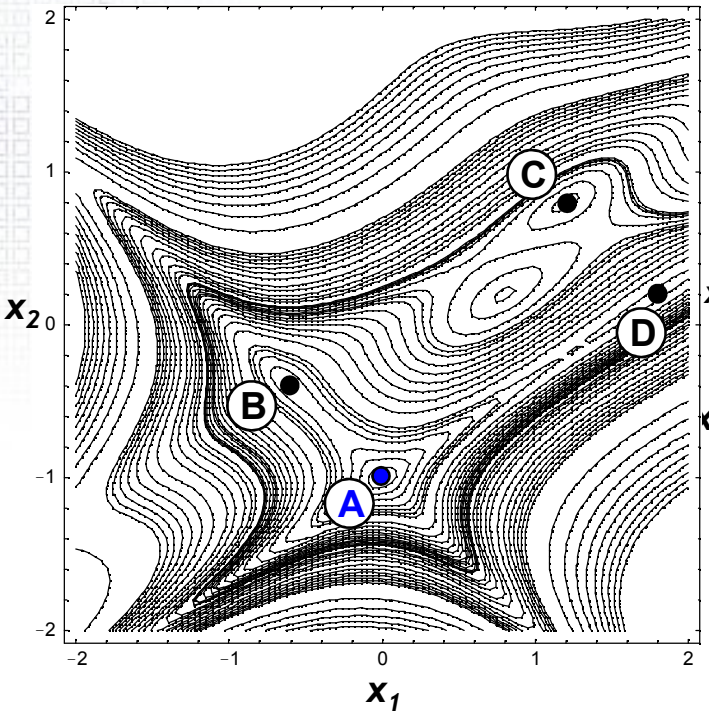
Goldstein-Price Function

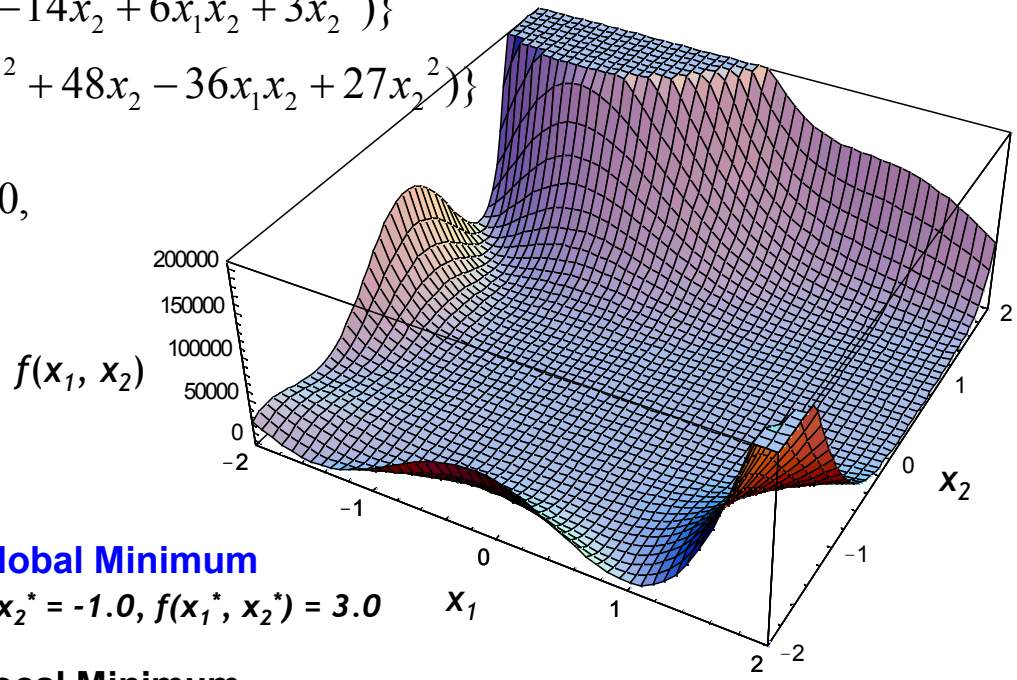
Minimize

$$f(x_1,x_2)=\{1+(x_1+x_2+1)^2\cdot(19-14x_1+3x_1^{\,2}-14x_2+6x_1x_2+3x_2^{\,2})\}$$
$$\cdot\{30+(2x_1-3x_2)^2\cdot(18-32x_1+12x_1^{\,2}+48x_2-36x_1x_2+27x_2^{\,2})\}$$

Subject to

$$g_1(x_1,x_2)=-2-x_1\le 0,\ g_2(x_1,x_2)=-2-x_2\le 0,$$
$$g_3(x_1,x_2)=x_1-2\le 0,\ g_4(x_1,x_2)=x_2-2\le 0$$





A : Global Minimum

$x_1^* = 0.0,\ x_2^* = -1.0,\ f(x_1^*, x_2^*) = 3.0$

B : Local Minimum

$x_1^* = -0.6,\ x_2^* = -0.4,\ f(x_1^*, x_2^*) = 30.0$

C : Local Minimum

$x_1^* = 1.2,\ x_2^* = 0.8,\ f(x_1^*, x_2^*) = 840.0$

D : Local Minimum

$x_1^* = 1.8,\ x_2^* = 0.2,\ f(x_1^*, x_2^*) = 84.0$

Simplex Class의 구현 예

```
class Simplex
{
public:
	Simplex();
	virtual ~Simplex();

	int m_nRowNo;			// Simplex Table의 행의 수(= 제약 조건식의 수)
	int m_nColumnNo;		// Simplex Table의 열의 수(= 원래의 설계 변수의 수 + 완화 변수의 수 + 잉여 변수의 수
	+ 인위 변수의 수)
	int m_nType;			// Phase I, II 여부

	int* m_pBDVId;			// 기저 변수들의 ID
	double** m_pA;			// 제약 조건식에 해당하는 계수들의 집합
	double* m_pB;			// 제약 조건식의 값들의 집합
	double* m_pC;			// 목적 함수식에 해당하는 계수들의 집합
	double* m_pW;			// 인위 목적 함수식의 설계 변수에 대한 계수들의 집합
	double m_fObj;			// 목적 함수값
	double m_fAObj;			// 인위 목적 함수값

	void InitializeSimplexTable();
	int FindPivotColumn();		// Pivot Column을 결정하는 함수
	int FindPivotRow();		// Pivot Row를 결정하는 함수
	int Pivot(int colNo, int rowNo);	// 주어진 Pivot Column과 Pivot Row로 Simplex Table의 Pivoting을 1번 수행하는 함수
	int CheckEndCondition();	// Simplex 방법의 종료 조건을 판단(목적 함수식 또는 인위 목적 함수식에 해당하는
	계수들이 모두 음이 아닌지 확인)하는 함수
	int Solve();			// Simplex 방법을 실행하는 함수
};
```

QP Class의 구현 예

```
class QP
{
public:
	QP();
	virtual ~QP();

	Simplex BXYD;			// 선형 방정식 "BX + Y = D"를 해결하기 위한 Simplex
	double** m_pH;			// H를 나타내는 2차원 배열
	double** m_pA;			// A를 나타내는 2차원 배열
	double** m_pN;			// N을 나타내는 2차원 배열
	double* m_pD;			// 선형 방정식을 푼 결과 Search Direction을 저장한 변수
	double* m_pU;			// 선형 방정식을 푼 결과 부등호 제약 조건식에 대한 Lagrange Multipliers를 저장한 변수
	double* m_pXi;			// 선형 방정식을 푼 결과 설계 변수의 양수 조건식에 대한 Lagrange Multipliers를 저장한
	변수
	double* m_pS;			// 선형 방정식을 푼 결과 부등호 제약 조건식에 대한 완화 변수를 저장한 변수
	double* m_pY;			// 선형 방정식을 푼 결과 등호 제약 조건식에 대한 Lagrange Multipliers를 저장한 변수
	double* m_pZ;			// 선형 방정식을 푼 결과 등호 제약 조건식에 대한 Lagrange Multipliers를 저장한 변수

	void ConstructSimplexTable();	// 선형 방정식 "BX + Y = D"에 해당하는 Simplex를 구성하는 함수
	int CheckEndCondition();	// QP 방법의 종료 조건을 판단(U*S = 0 & Xi*D = 0)하는 함수
	int Solve();			// QP를 실행하는 함수
};
```

SQP-CSD Class의 구현 예

```
class SQPCSD
{
public:
        SQPCSD();
        virtual ~SQPCSD();

        int m_nDVNo;                                        // 설계 변수의 수
        int m_nCFNo;                                        // 제약 조건의 수
        int m_nIterationNo;                                 // 반복 회수(k)
        double m_nStepSize;                                 // 임시 이동 거리(t)
        double m_nPenaltyParameter;                         // 벌칙 매개 변수(R)
        double* DV;                                         // 설계 변수들을 저장한 배열

        double ObjectiveFunction(int DVNo, double* DV);     // 목적 함수식
        void Constraints(int CFNo, double* CF);             // 제약 조건식
        void OFGradients(double* OFG);                      // 목적 함수식에 대한 Gradient Vector
        void CFGradients(double** CFG);                     // 제약 조건식에 대한 Gradient Vector

        int DetermineSearchDirection();                     // SPQ 방법에 따라 탐색 방향을 결정하는 함수
        double CalculatePenaltyParameter();                 // 벌칙 매개 변수(R)를 계산하는 함수
        double CalculateMaxConstraintsViolation()           // 제약 조건의 최대 위배 값(V)을 결정하는 함수
        double ConstructDescentFunction();                  // 강하 함수를 구성하는 함수
        int DetermineStepSize();                            // CSD 방법에 따라 이동 거리를 결정하는 함수
        int CheckEndCondition();                            // SQP-CSD 방법의 종료 조건을 판단
        int Solve();                                        // SQP-CSD 방법을 실행하는 함수
};
```

CSD Programming Guide

- Simplex 방법을 이용한 2차 계획 문제의 풀이 방법 (1)

Kuhn-Tucker 필요 조건: $\nabla L = \mathbf{0}$

$$\frac{\partial L}{\partial \mathbf{d}_{(n\times 1)}} = \mathbf{c}_{(n\times 1)} + \mathbf{H}_{(n\times n)}\mathbf{d}_{(n\times 1)} + \mathbf{A}_{(n\times m)}\mathbf{u}_{(m\times 1)} + \mathbf{N}_{(n\times p)}\mathbf{v}_{(p\times 1)} = \mathbf{0}$$

$$\frac{\partial L}{\partial \mathbf{u}_{(m\times 1)}} = \mathbf{A}^{T}{}_{(m\times n)}\mathbf{d}_{(n\times 1)} + \mathbf{s}_{(m\times 1)} - \mathbf{b}_{(m\times 1)} = \mathbf{0}$$

$$\frac{\partial L}{\partial \mathbf{v}_{(p\times 1)}} = \mathbf{N}^{T}{}_{(p\times n)}\mathbf{d}_{(n\times 1)} - \mathbf{e}_{(p\times 1)} = \mathbf{0}$$

→ 이 식들은 설계 변수 d에 대해 모두 선형이므로,
이 식들로부터 설계 변수 d를 구하는 문제는 등호 제약 조건만으로 이루어진 선형 계획 문제임

여기서, $u_i, s_i \geq 0;\ i = 1\ to\ m$

$u_i s_i = 0; i = 1\ to\ m$

→ 선형 부정 방정식으로부터 구한 해가 이 비선형 부정 방정식을 만족하는지 확인하여 해를 확정한다.

설계 변수($d^{(0)}$)가 부호에 제한이 없기 때문에 문제를 풀기 전에 먼저 설계 변수를 아래와 같이 변환해야 함

$$\mathbf{d}_{(n\times 1)} = \mathbf{d}_{(n\times 1)}^{+} - \mathbf{d}_{(n\times 1)}^{-} \qquad \text{여기서, } \mathbf{d}_{(n\times 1)}^{+},\ \mathbf{d}_{(n\times 1)}^{-} \geq 0$$

등호 제약 조건에 대한 Lagrange multiplier $\mathbf{v}_{(p\times 1)}$도 부호의 제한이 없으므로 다음과 같이 변환해야 함

$$\mathbf{v}_{(p\times 1)} = \mathbf{y}_{(p\times 1)} - \mathbf{z}_{(p\times 1)}$$

CSD Programming Guide

- Simplex 방법을 이용한 2차 계획 문제의 풀이 방법 (2)

Kuhn-Tucker 필요 조건: $\nabla L = \mathbf{0}$

$$\frac{\partial L}{\partial \mathbf{d}_{(n\times 1)}} = \mathbf{c}_{(n\times 1)} + \mathbf{H}_{(n\times n)}\mathbf{d}_{(n\times 1)} + \mathbf{A}_{(n\times m)}\mathbf{u}_{(m\times 1)} + \mathbf{N}_{(n\times p)}\mathbf{v}_{(p\times 1)} = \mathbf{0}$$

$$\frac{\partial L}{\partial \mathbf{u}_{(m\times 1)}} = \mathbf{A}^{T}{}_{(m\times n)}\mathbf{d}_{(n\times 1)} + \mathbf{s}_{(m\times 1)} - \mathbf{b}_{(m\times 1)} = \mathbf{0} \qquad u_i s_i = 0; i = 1 \, to \, m$$

$$\frac{\partial L}{\partial \mathbf{v}_{(p\times 1)}} = \mathbf{N}^{T}{}_{(p\times n)}\mathbf{d}_{(n\times 1)} - \mathbf{e}_{(p\times 1)} = \mathbf{0}$$

여기서, $u_i, s_i \geq 0; \; i = 1 \, to \, m$

Kuhn-Tucker 필요 조건(행렬식 표현)

$$\underbrace{\begin{bmatrix} \mathbf{H}_{(n\times n)} & -\mathbf{H}_{(n\times n)} & \mathbf{A}_{(n\times m)} & \mathbf{0}_{(n\times m)} & \mathbf{N}_{(n\times p)} & -\mathbf{N}_{(n\times p)} \\ \mathbf{A}^{T}{}_{(m\times n)} & -\mathbf{A}^{T}{}_{(m\times n)} & \mathbf{0}_{(m\times m)} & \mathbf{I}_{(m\times m)} & \mathbf{0}_{(m\times p)} & \mathbf{0}_{(m\times p)} \\ \mathbf{N}^{T}{}_{(p\times n)} & -\mathbf{N}^{T}{}_{(p\times n)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times p)} & \mathbf{0}_{(p\times p)} \end{bmatrix}}_{= \mathbf{B}_{((n+m+p)\times(2n+2m+2p))}} \underbrace{\begin{bmatrix} \mathbf{d}^{+}{}_{(n\times 1)} \\ \mathbf{d}^{-}{}_{(n\times 1)} \\ \mathbf{u}_{(m\times 1)} \\ \mathbf{s}_{(m\times 1)} \\ \mathbf{y}_{(p\times 1)} \\ \mathbf{z}_{(p\times 1)} \end{bmatrix}}_{= \mathbf{X}_{((2n+2m+2p)\times 1)}} = \underbrace{\begin{bmatrix} -\mathbf{c}_{(n\times 1)} \\ \mathbf{b}_{(m\times 1)} \\ \mathbf{e}_{(p\times 1)} \end{bmatrix}}_{= \mathbf{D}_{((n+m+p)\times 1)}}$$

$u_i s_i = 0; i = 1 \, to \, m$

$$\mathbf{B}_{((n+m+p)\times(2n+2m+2p))}\mathbf{X}_{((2n+2m+2p)\times 1)} = \mathbf{D}_{((n+m+p)\times 1)}$$

CSD Programming Guide
- CSD 방법 요약

① 목적함수는 2차 형식, 제약 조건은 1차형식으로 근사화 (2차 계획 문제로 근사화)

② Lagrange 함수와 쿤-터커 조건을 사용하여 행렬로 표현

$$\begin{bmatrix} \mathbf{H}_{(n\times n)} & -\mathbf{H}_{(n\times n)} & \mathbf{A}_{(n\times m)} & \mathbf{0}_{(n\times m)} & \mathbf{N}_{(n\times p)} & -\mathbf{N}_{(n\times p)} \\ \mathbf{A}^T{}_{(m\times n)} & -\mathbf{A}^T{}_{(m\times n)} & \mathbf{0}_{(m\times m)} & \mathbf{I}_{(m\times m)} & \mathbf{0}_{(m\times p)} & \mathbf{0}_{(m\times p)} \\ \mathbf{N}^T{}_{(p\times n)} & -\mathbf{N}^T{}_{(p\times n)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times p)} & \mathbf{0}_{(p\times p)} \end{bmatrix} \begin{bmatrix} \mathbf{d}^+{}_{(n\times 1)} \\ \mathbf{d}^-{}_{(n\times 1)} \\ \mathbf{u}_{(m\times 1)} \\ \mathbf{s}_{(m\times 1)} \\ \mathbf{y}_{(p\times 1)} \\ \mathbf{z}_{(p\times 1)} \end{bmatrix} = \begin{bmatrix} -\mathbf{c}_{(n\times 1)} \\ \mathbf{b}_{(m\times 1)} \\ \mathbf{e}_{(p\times 1)} \end{bmatrix}$$

$$\text{Minimize } \bar{f} = \mathbf{c}^T{}_{(1\times n)}\mathbf{d}_{(n\times 1)} + \frac{1}{2}\mathbf{d}^T{}_{(1\times n)}\mathbf{d}_{(n\times 1)}$$

$$\text{Subject to } \mathbf{N}^T{}_{(p\times n)}\mathbf{d}_{(n\times 1)} = \mathbf{e}_{(p\times 1)}$$

$$\mathbf{A}^T{}_{(m\times n)}\mathbf{d}_{(n\times 1)} \le \mathbf{b}_{(m\times 1)}$$

③ Simplex Method를 사용하여 탐색방향 찾음

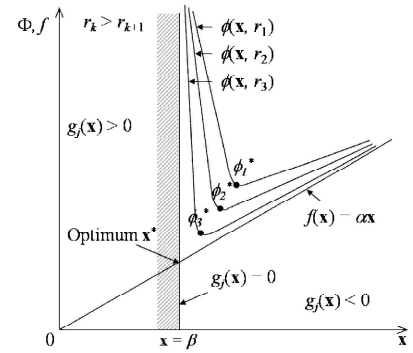
1

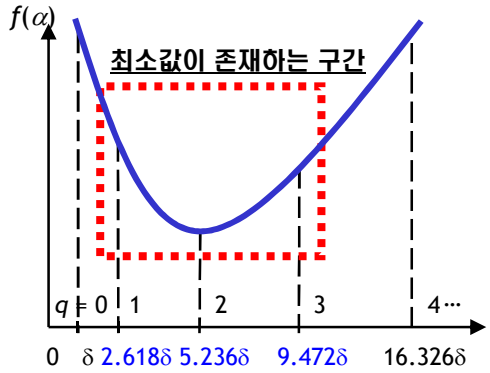
	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
Y1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
Y2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
X8	1/3	1/3	-1/3	-1/3	0	0	0	1	0	0	0	0	1	0	0	2/3	-
Y4	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1	1
Y5	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	-
A. Obj.	0	0	0	0	-2/3	1	1	0	-1	-1	0	0	1	0	0	w-4	-

2

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
Y1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
Y2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
X8	1/3	1/3	-1/3	-1/3	0	0	0	1	0	0	0	0	1	0	0	2/3	-
X9	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1	-
Y5	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1
A. Obj.	-1	0	1	0	-2/3	1	1	0	0	-1	0	0	1	1	0	w-3	-

④ 강하 함수를 목적함수로 사용하여 1차원 탐색 방법 (예: 황금 분할법)을 이용하여 탐색 방향으로 이동거리 계산





CSD Programming Guide
- CSD Class 예시

```
#include<vector>
#include "Matrix.h“
#include “Simplex.h“
using namespace std;

typedef double(*FP)(double *x);

class CSD
{
public:
    CSD();                  //생성자
    virtual ~CSD();         //소멸자

//Member Variables
    FP m_ObjFn;                         //목적함수
    std::vector<FP> m_EqualFn;          //등호제약조건
    std::vector<FP> m_UnequalFn;        //부등호제약조건

    int NumOfEqual;         //등식의 개수
    int NumOfUnEqual;       //부등식의 개수
    int NumOfVariable;      //미지수의 개수

    double *m_x;                //점의좌표
    double *m_d;                //탐색방향
    double m_MinValue;          //최소값
    Matrix m_SimplexTable;  //쿤-터커 필요조건 행렬
```

```
//Member Function

// 초기 시작 위치와 변수의 개수 입력
    void InitializeCSD(double *_x,int n);
//목적함수, 등호 및 부등호 제약조건 입력
    void AddObjectFunction(double (*f)(double*));
    void AddEqualConstraint(double (*f)(double*));
    void AddUnEqualConstraintFn(double (*f)(double*));

    void Solve();

//쿤-터커 필요조건 행렬식 만듦
    void BuildSimplexTable();
//탐색방향 계산
    void FindSearchDirection();
//End Condition Check
    bool CheckEndCondition();
//탐색 방향으로의 최소값과 최소점 계산
    void FindNextMinPoint();
//강하 함수(Penalty Function)을 정의
    static double PenaltyFunction(double *_x);
};
```

class 멤버 함수를 함수 포인터로 전달하기 위하여 static으로 선언함

CSD Programming Guide

1) 시작점의 좌표, 목적 함수, 등호/부등호 제약 조건 입력

(Example)

```
double ObjectFn(double *x);
double UnequalConst1(double *x);
double UnequalConst2(double *x);
double EqualConst1(double *x);

int main()
{
    CSD csd1;

    int n = 2;
    double *x = new double [n];
    x[0]=1;
    x[1]=1;

    csd1.InitializeCSD(x,n);
    csd1.AddObjectFunction(ObjectFn);
    csd1.AddUnequalConstraint(UnequalConst1);
    csd1.AddUnequalConstraint(UnequalConst2);
    csd1.AddEqualConstraint(EqualConst1);
    csd1.Solve();

    delete[] x;
}
```

변수 개수 2개

시작 위치 설정 (1,1)

CSD 초기화

목적함수, 등호 및 부등호 제약조건 입력

CSD Programming Guide
2) Solve 함수의 동작

```
void CSD::Solve()
{
    while(1)
    {
        BuildSimplexTable();

        FindSearchDirection();

        if( CheckEndCondition() )
            break;

        FindNextMinPoint();
    }
}
```

① 목적함수는 2차, 제약 조건은 1차로 근사화 (2차 계획 문제로 표현)

$$\text{Minimize } \bar{f} = \mathbf{c}^T_{(1\times n)}\mathbf{d}_{(n\times 1)} + \frac{1}{2}\mathbf{d}^T_{(1\times n)}\mathbf{d}_{(n\times 1)} \quad \text{Subject to } \mathbf{N}^T_{(p\times n)}\mathbf{d}_{(n\times 1)} = \mathbf{e}_{(p\times 1)},\ \mathbf{A}^T_{(m\times n)}\mathbf{d}_{(n\times 1)} \le \mathbf{b}_{(m\times 1)}$$

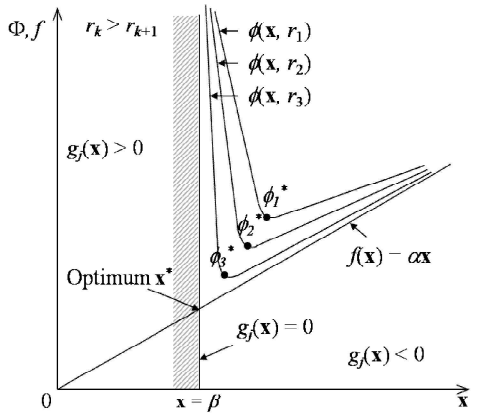
② Lagrange 함수와 쿤-터커 조건을 사용하여 행렬로 표현

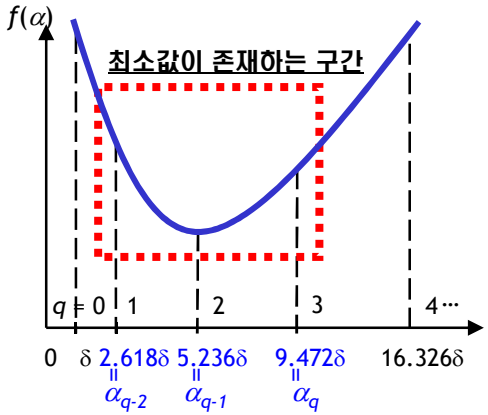
$$\begin{bmatrix} \mathbf{H}_{(n\times n)} & -\mathbf{H}_{(n\times n)} & \mathbf{A}_{(n\times m)} & \mathbf{0}_{(n\times m)} & \mathbf{N}_{(n\times p)} & -\mathbf{N}_{(n\times p)} \\ \mathbf{A}^T_{(m\times n)} & -\mathbf{A}^T_{(m\times n)} & \mathbf{0}_{(m\times m)} & \mathbf{I}_{(m\times m)} & \mathbf{0}_{(m\times p)} & \mathbf{0}_{(m\times p)} \\ \mathbf{N}^T_{(p\times n)} & -\mathbf{N}^T_{(p\times n)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times p)} & \mathbf{0}_{(p\times p)} \end{bmatrix} \begin{bmatrix} \mathbf{d}^+_{(n\times 1)} \\ \mathbf{d}^-_{(n\times 1)} \\ \mathbf{u}_{(m\times 1)} \\ \mathbf{s}_{(m\times 1)} \\ \mathbf{y}_{(p\times 1)} \\ \mathbf{z}_{(p\times 1)} \end{bmatrix} = \begin{bmatrix} -\mathbf{c}_{(n\times 1)} \\ \mathbf{b}_{(m\times 1)} \\ \mathbf{e}_{(p\times 1)} \end{bmatrix}$$

③ Simplex Method를 사용하여 탐색방향 찾음

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	Y1	Y2	Y3	Y4	Y5	bi	bi/ai
Y1	1	0	-1	0	1/3	-1	0	0	0	0	1	0	0	0	0	1	-
Y2	0	1	0	-1	1/3	0	-1	0	0	0	0	1	0	0	0	1	-
X8	1/3	1/3	-1/3	-1/3	0	0	0	1	0	0	0	0	1	0	0	2/3	-
Y4	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1	1
Y5	0	-1	0	1	0	0	0	0	0	1	0	0	0	0	1	1	-
A. Obj.	0	0	0	0	-2/3	1	1	0	-1	-1	0	0	1	0	0	w-4	-

④ 강하 함수를 목적함수로 사용하여 1차원 탐색 방법 (예: 황금 분할법)을 이용하여 탐색 방향으로 이동거리 계산





CSD Programming Guide
3) Simplex Table 구성

n : 변수의 개수(m_NumOfVariable)
m : 부등호 제약조건 식의 개수(m_NumOfUnequal)
p : 등호 제약조건 식의 개수(m_NumOfEqual)

Simplex Table 구성: BuildSimplexTable 함수 구현

주어진 목적함수와 제약 조건 식으로부터 다음 매트릭스를 구성해야 함

$$\begin{bmatrix} \mathbf{H}_{(n\times n)} & -\mathbf{H}_{(n\times n)} & \mathbf{A}_{(n\times m)} & \mathbf{0}_{(n\times m)} & \mathbf{N}_{(n\times p)} & -\mathbf{N}_{(n\times p)} \\ \mathbf{A}^T{}_{(m\times n)} & -\mathbf{A}^T{}_{(m\times n)} & \mathbf{0}_{(m\times m)} & \mathbf{I}_{(m\times m)} & \mathbf{0}_{(m\times p)} & \mathbf{0}_{(m\times p)} \\ \mathbf{N}^T{}_{(p\times n)} & -\mathbf{N}^T{}_{(p\times n)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times p)} & \mathbf{0}_{(p\times p)} \end{bmatrix} = \mathbf{B}_{((n+m+p)\times(2n+2m+2p))}$$

문제로부터 구할 수 있는 것

탐색 방향
(부호 제약이 없으므로 두 개로 분리)

Lagrange multipliers

$$\mathbf{B}_{((n+m+p)\times(2n+2m+2p))}\mathbf{X}_{((2n+2m+2p)\times 1)} = \mathbf{D}_{((n+m+p)\times 1)}$$

$$\begin{bmatrix} \mathbf{d}^+_{(n\times 1)} \\ \mathbf{d}^-_{(n\times 1)} \\ \mathbf{u}_{(m\times 1)} \\ \mathbf{s}_{(m\times 1)} \\ \mathbf{y}_{(p\times 1)} \\ \mathbf{z}_{(p\times 1)} \end{bmatrix} = \begin{bmatrix} -\mathbf{c}_{(n\times 1)} \\ \mathbf{b}_{(m\times 1)} \\ \mathbf{e}_{(p\times 1)} \end{bmatrix} = \mathbf{D}_{((n+m+p)\times 1)}$$

완화 변수

문제로부터 구할 수 있는 것

$$= \mathbf{X}_{((2n+2m+2p)\times 1)}$$

구하려고 하는 것

목적 함수의 Gradient Vector

$$\mathbf{H}_{(n\times n)} = \mathbf{I}_{(n\times n)},\quad \mathbf{c}_{(n\times 1)} = \begin{bmatrix} \partial f(\mathbf{x})/\partial x_1 \\ \vdots \\ \partial f(\mathbf{x})/\partial x_n \end{bmatrix},\quad \mathbf{b}_{(m\times 1)} = \begin{bmatrix} -g_1(\mathbf{x}) \\ \vdots \\ -g_m(\mathbf{x}) \end{bmatrix},\quad \mathbf{e}_{(p\times 1)} = \begin{bmatrix} -h_1(\mathbf{x}) \\ \vdots \\ -h_p(\mathbf{x}) \end{bmatrix}$$

부등호 제약 조건의 Gradient Vector

$$\mathbf{A}_{(n\times m)} = \begin{bmatrix} \partial g_1(\mathbf{x})/\partial x_1 & \cdots & \partial g_m(\mathbf{x})/\partial x_1 \\ \vdots & \vdots & \vdots \\ \partial g_1(\mathbf{x})/\partial x_n & \cdots & \partial g_m(\mathbf{x})/\partial x_n \end{bmatrix},$$

등호 제약 조건의 Gradient Vector

$$\mathbf{N}_{(n\times p)} = \begin{bmatrix} \partial h_1(\mathbf{x})/\partial x_1 & \cdots & \partial h_p(\mathbf{x})/\partial x_1 \\ \vdots & \vdots & \vdots \\ \partial h_1(\mathbf{x})/\partial x_n & \cdots & \partial h_p(\mathbf{x})/\partial x_n \end{bmatrix}$$

CSD Programming Guide
3) Simplex Table 구성

Simplex Table 구성: BuildSimplexTable 함수 구현

2변수 함수 $f(\mathbf{x}) = f(x_1, x_2)$ 의 Gradient Vector($\nabla f(\mathbf{x})$)는 다음과 같은 Central Difference Method을 이용하여 수치적으로 계산할 수 있음

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial f(x_1, x_2)}{\partial x_1} \\ \dfrac{\partial f(x_1, x_2)}{\partial x_2} \end{bmatrix}$$

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = \frac{f(x_1 + \Delta x_1, x_2) - f(x_1 - \Delta x_1, x_2)}{(x_1 + \Delta x_1) - (x_1 - \Delta x_1)} = \frac{f(x_1 + \Delta x_1, x_2) - f(x_1 - \Delta x_1, x_2)}{2\Delta x_1}$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} = \frac{f(x_1, x_2 + \Delta x_2) - f(x_1, x_2 - \Delta x_2)}{(x_2 + \Delta x_2) - (x_2 - \Delta x_2)} = \frac{f(x_1, x_2 + \Delta x_2) - f(x_1, x_2 - \Delta x_2)}{2\Delta x_2}$$

여기서, $\Delta x_1, \Delta x_2$ 은 아주 작은 양수(예, 10^{-6})

CSD Programming Guide
3) Simplex Table 구성

n : 변수의 개수(m_NumOfVariable)
m : 부등호 제약조건 식의 개수(m_NumOfUnequal)
p : 등호 제약조건 식의 개수(m_NumOfEqual)

Simplex Table 구성: BuildSimplexTable 함수 구현

주어진 목적함수와 제약 조건 식으로부터
다음 매트릭스를 구성해야 함

$$\begin{bmatrix} \mathbf{H}_{(n\times n)} & -\mathbf{H}_{(n\times n)} & \mathbf{A}_{(n\times m)} & \mathbf{0}_{(n\times m)} & \mathbf{N}_{(n\times p)} & -\mathbf{N}_{(n\times p)} \\ \mathbf{A}^T{}_{(m\times n)} & -\mathbf{A}^T{}_{(m\times n)} & \mathbf{0}_{(m\times m)} & \mathbf{I}_{(m\times m)} & \mathbf{0}_{(m\times p)} & \mathbf{0}_{(m\times p)} \\ \mathbf{N}^T{}_{(p\times n)} & -\mathbf{N}^T{}_{(p\times n)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times p)} & \mathbf{0}_{(p\times p)} \end{bmatrix} \begin{bmatrix} \mathbf{d}^+{}_{(n\times 1)} \\ \mathbf{d}^-{}_{(n\times 1)} \\ \mathbf{u}_{(m\times 1)} \\ \mathbf{s}_{(m\times 1)} \\ \mathbf{y}_{(p\times 1)} \\ \mathbf{z}_{(p\times 1)} \end{bmatrix} = \begin{bmatrix} -\mathbf{c}_{(n\times 1)} \\ \mathbf{b}_{(m\times 1)} \\ \mathbf{e}_{(p\times 1)} \end{bmatrix}$$

$$\mathbf{B}_{((n+m+p)\times(2n+2m+2p))}\mathbf{X}_{((2n+2m+2p)\times 1)} = \mathbf{D}_{((n+m+p)\times 1)}$$

(n+m+p)개의 등식으로만 구성된 연립 일차 방정식 → 미지수의 개수 $(2n+2m+2p)$ > 식의 개수 $(n+m+p)$

$$\mathbf{B}_{((n+m+p)\times(2n+2m+2p))}\mathbf{X}_{((2n+2m+2p)\times 1)} + \mathbf{Y}_{(n+m+p)} = \mathbf{D}_{((n+m+p)\times 1)}$$

식의 개수만큼 인위 변수(artificial variable) 추가

인위 목적 함수(artificial object function) 추가

CSD Programming Guide
3) Simplex Table 구성

n : 변수의 개수(m_NumOfVariable)
m : 부등호 제약조건 식의 개수(m_NumOfUnequal)
p : 등호 제약조건 식의 개수(m_NumOfEqual)

Simplex Table 구성: BuildSimplexTable 함수 구현

$$\mathbf{B}_{((n+m+p)\times(2n+2m+2p))}\mathbf{X}_{((2n+2m+2p)\times 1)} + \mathbf{Y}_{(n+m+p)} = \mathbf{D}_{((n+m+p)\times 1)}$$

$$\begin{bmatrix}
\mathbf{H}_{(n\times n)} & -\mathbf{H}_{(n\times n)} & \mathbf{A}_{(n\times m)} & \mathbf{0}_{(n\times m)} & \mathbf{N}_{(n\times p)} & -\mathbf{N}_{(n\times p)} & \mathbf{I}_{(n\times n)} & \mathbf{0}_{(n\times m)} & \mathbf{0}_{(n\times p)} & -\mathbf{c}_{(n\times 1)} \\
\mathbf{A}^{T}_{(m\times n)} & -\mathbf{A}^{T}_{(m\times n)} & \mathbf{0}_{(m\times m)} & \mathbf{I}_{(m\times m)} & \mathbf{0}_{(m\times p)} & \mathbf{0}_{(m\times p)} & \mathbf{0}_{(m\times n)} & \mathbf{I}_{(m\times m)} & \mathbf{0}_{(m\times p)} & \mathbf{b}_{(m\times 1)} \\
\mathbf{N}^{T}_{(p\times n)} & -\mathbf{N}^{T}_{(p\times n)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times m)} & \mathbf{0}_{(p\times p)} & \mathbf{0}_{(p\times p)} & \mathbf{0}_{(p\times n)} & \mathbf{0}_{(p\times m)} & \mathbf{I}_{(p\times p)} & \mathbf{e}_{(p\times 1)}
\end{bmatrix}$$

Artificial Object Function
(모든 열의 합에 (-)를 붙여 구함)

Matrix
m_SimplexTable;

행의 개수 $(n+m+p)+1$

인위목적함수

열의 개수 $(3n+3m+3p)+1 = 3(n+m+p)+1$

등호의 우변

CSD Programming Guide
3) Simplex Table 구성

Simplex Table 구성: BuildSimplexTable 함수 구현

- 함수 구현 시 주의 사항

✓ 인위 목적 함수는 각 열을 모두 더한 다음 (-)를 붙여주면 된다.

✓ 인위 목적 함수를 만들기 전에 반드시 가장 마지막 열의 부호가 모두 양수 또는 0인지 확인하고, 음수일 경우 행 전체에 (-)를 곱해 준다.

✓ m_SimplexTable을 구성할 때, Matrix class의 SetPartOfMatrix 함수를 사용하면 편리하다.

CSD Programming Guide
4) 탐색 방향을 찾음

탐색 방향을 찾음: FindSearchDirection **함수 구현**

variableTable**과** biaiTable**을 선언**

Simplex class**의** Solve **함수 실행**

모든 $u_i s_i = 0$ **인가?**

Yes

Simplex **종료**

No

저장된 Simplex**를 불러와 다시 실행**

Matrix
VariableTable;

Matrix
biaiTable;

Matrix
m_SimplexTable;

행의 개수
$(n+m+p)+1$

열의 개수 $3(n+m+p)+1$

※ 주의 사항

Pivot **열 또는 행을 찾을 때, 선택할 수 있는 열 또는 행이 여러 개일 수 있다.**

이 경우, 선택하지 않은 열 또는 행 정보를 가지는 Simplex**를 저장하고 있어야 한다.**

CSD Programming Guide
4) 탐색 방향을 찾음

탐색 방향을 찾음: FindSearchDirection 함수 구현

```
void CSD::FindSearchDirection()
{
    ……
    Simplex mySimplex;
    mySimplex.Initialize(…);

    while(1)
    {
        //인위목적함수가 0이되는 쌍이
        //있으면 true를 return
        //없으면 false를 return
        bool isFound = mySimplex.Solve();

        if(isFound)
        {
            uisi =0 인지 확인후 참이면 종료
        }

        //저장된 Simplex를 하나 불러옴
        mySimplex = …
    }
}
```

```
bool Simplex::Solve()
{
    while(1)
    {
        if (m_pivotColumn == -1)
            FindPivotColumn();
        if (m_pivotColumn == -1)
        {
            return false;
        }
        if (m_pivotRow == -1)
            FindPivotRow();
        if (m_pivotRow == -1)
        {
            return false;
        }
        Pivot();
        if(CheckEndCondition())
            return true;
    }
    return true;
}
```

최소값이 음수가 아님
(모든 계수가 양수)

bi/ai 비율이
최소인 양수가 없음

CSD Programming Guide
4) 탐색 방향을 찾음

탐색 방향을 찾음: FindSearchDirection 함수 구현

- ✓ 반드시 Roll-Back 할 수 있도록 구현되어야 함
- ✓ Roll-back 기능은 각자 방법대로 구현 가능. 위의 예제는 참고 사항
- ✓ 탐색 방향은 $\mathbf{d}_{(n\times 1)} = \mathbf{d}^{+}{}_{(n\times 1)} - \mathbf{d}^{-}{}_{(n\times 1)}$ 로 구해짐

CSD Programming Guide
5) 강하 함수 구현

✓ **강하 함수의 정의**

$$\Phi(\mathbf{x}) = f(\mathbf{x}) + R \cdot V(\mathbf{x})$$

여기서, $R = \max\{R_0, |\mathbf{v}| + \mathbf{u}\}$: **벌칙 매개 변수로서 모든** Lagrange multiplier**의 합**

$V(\mathbf{x}) = \max\{0; |\mathbf{h}|; \mathbf{g}\}$: **현재의 설계점에서의 최대 제약 조건 위배 값**

```
double CSD::PenaltyFunction(double *_x)
{
    // Simplex의 계산 결과에서 구한  v, u 를 더함  R
    // _x값을 대입하여 등호 및 부등호 제약 조건의 함수값을 더함  V(x)
    return m_ObjFn(_x) + R * V;
}
```

CSD Programming Guide

6) 1차원 탐색 방법(예: 황금분할법)을 이용하여 강하 함수(Penalty Function)를 최소로 하는 이동거리를 검출

FindNextMinPoint 함수 구현

```
void CSD::FindNextMinPoint()
{
    double** section;
    double* section = new double* [3];
    for(int i=0;i<3;i++)
    {
        section[i] = new double [m_NumOfVariable];
    }

    findMinValueExistSection(m_X, m_d, section, PenaltyFunction);
    m_MinValue = GoldenSectionSearch(section, PenaltyFunction, m_X);

    for(int i=0;i<3;i++)
    {
        delete[] section[i];
    }
    delete []section;
}
```

QP에서 구해진 방향에 따라
1차원 탐색 방법(예: 황금분할법)을 이용하여
강하 함수(Penalty Function)를
최소로 하는 이동거리를 검출 함

Ch6. 제약 비선형 최적화 프로그램 소개

6.1 제약 비선형 최적화 프로그램의 구성

6.2 제약 비선형 최적화 프로그램의 사용 방법

6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예

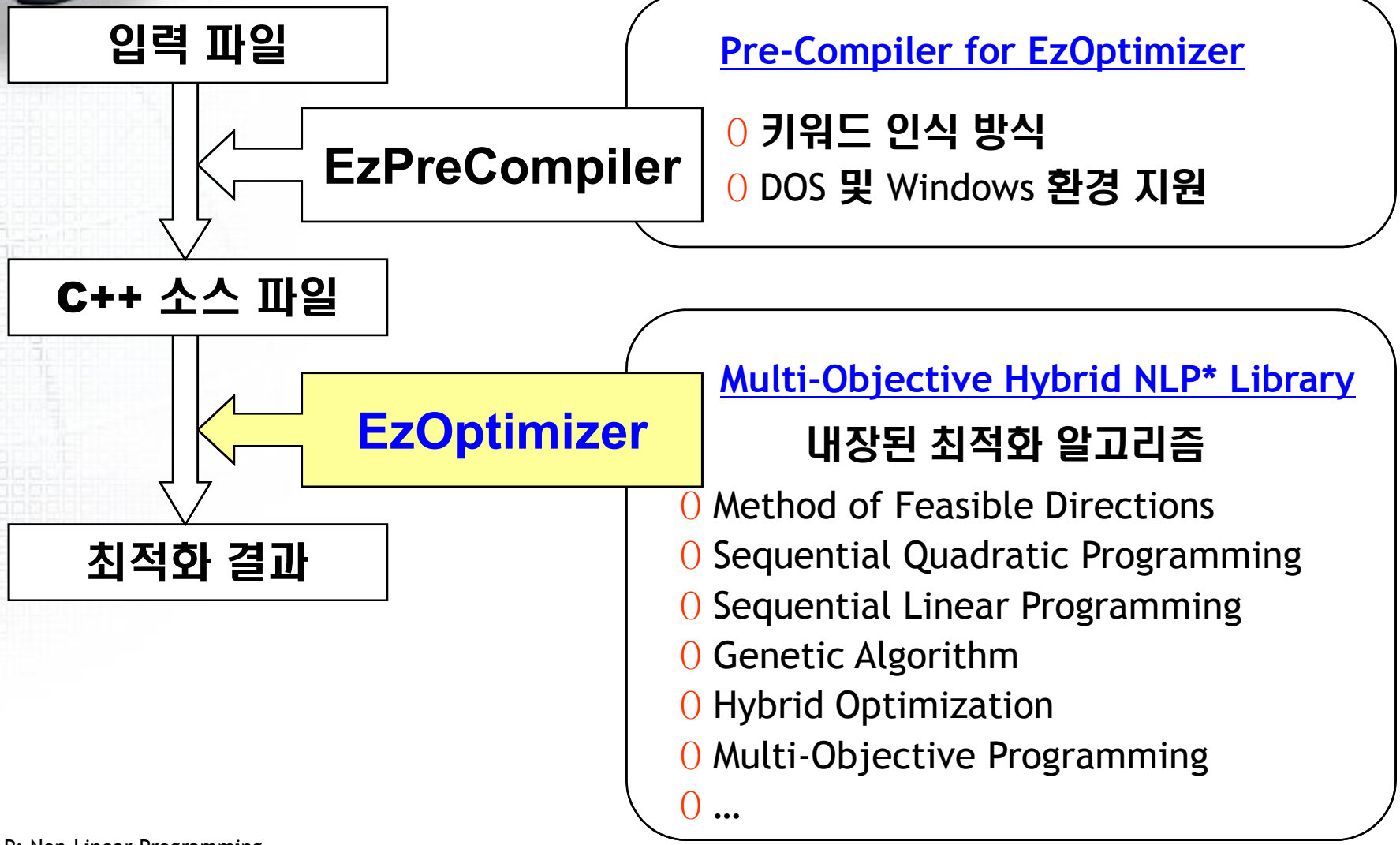
Advanced
Ship
Design
Automation
Laboratory

6.1 제약 비선형 최적화 프로그램의 구성(1)

- **비선형 최적화 라이브러리 'EzOptimizer'**
  - **다양한 최적화 알고리즘을 포함**
  - **C++ 언어로 구현**
  - **최적화 모듈을 사용자의 프로그램에 쉽게 이식 가능**
  - **최적화 과정을 제어할 수 있는 다양한 옵션 지정 가능**
  - **헤더 파일**(EzOptimizer.h)**과 라이브러리 파일**(EzOptimizerLibraryD.lib/EzOptimizerLibraryR.lib)**로 구성**

- **'EzOptimizer'를 위한 프리-컴파일러 'EzPreCompiler'**
  - **'EzOptimizer'의 보다 편리한 사용을 위해 개발**
  - **키워드**(Keyword) **인식 방식**
  - **'EzPreCompile'용 입력 파일을 C++ 소스 파일로 변환**
  - **특정 키워드에 의한 다중 최적화 블록도 처리 가능**
  - **DOS 및 Windows 버전 개발**

6.1 제약 비선형 최적화 프로그램의 구성(2)



* NLP: Non-Linear Programming
* 1: **노명일, 혼동 최적화 방법에 의한 다분야 최적화 기법에 관한 연구, 서울대학교 조선해양공학과 석사학위논문**, 2000.2
* 2: Kyu-Yeul Lee, Seon-Ho Cho, Myung-Il Roh, “An Efficient Global-Local Hybrid Optimization Method Using Design Sensitivity Analysis”, International Journal of Vehicle Design(SCIE), Vol. 28, No. 4, pp.300~317, 2002.7

6.2 제약 비선형 최적화 프로그램의 사용 방법

1. 사용자가 작성하는 프로그램의 디렉토리에 "EzOptimizer.h"와 "EzOptimizerD.lib"(또는 "EzOptimizerR.lib")를 복사

2. 사용자가 작성하는 프로그램의 Source Code 상단에 다음과 같은 내용을 추가

   #include "EzOptimizer.h"

3. EzOptimizer를 사용하는 프로그램 작성

4. 사용자가 작성하는 프로그램의 프로젝트 파일에 "EzOptimizerD.lib"(또는 "EzOptimizerR.lib")를 포함

5. 사용자가 작성한 프로그램을 컴파일, 링크 후 실행

6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예 - 수학 최적화 문제

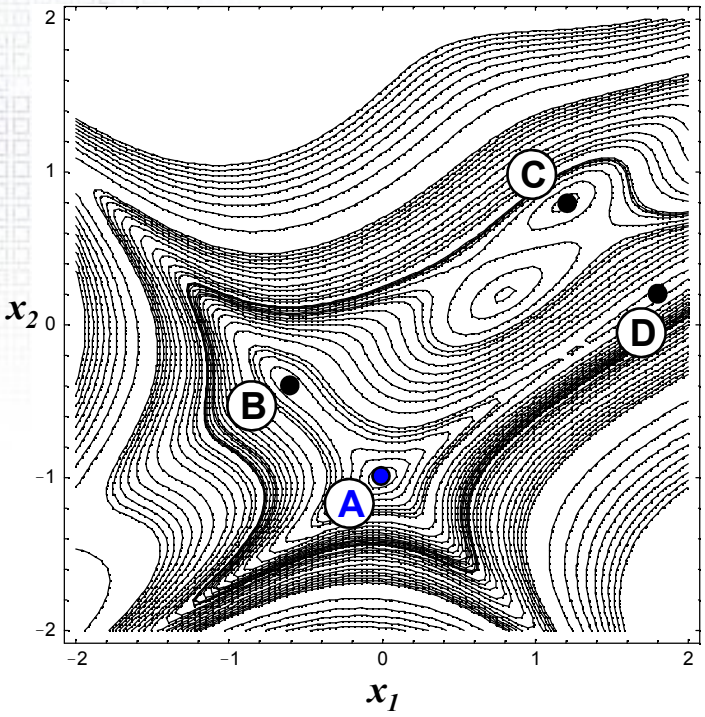
Goldstein-Price Function

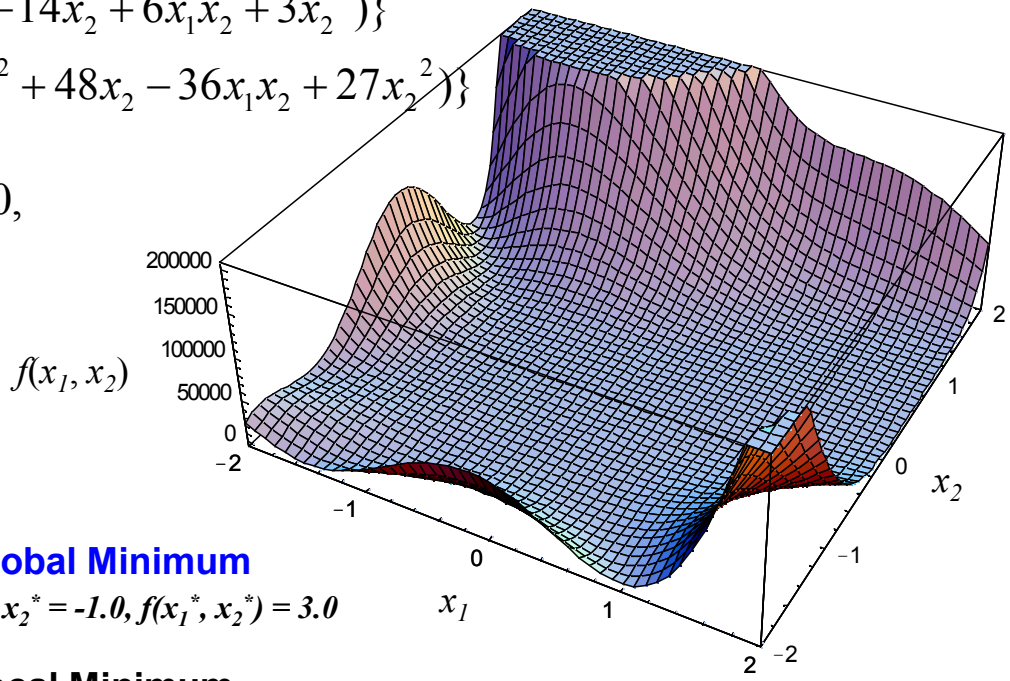
Minimize

$$f(x_1,x_2)=\{1+(x_1+x_2+1)^2\cdot(19-14x_1+3x_1^{\,2}-14x_2+6x_1x_2+3x_2^{\,2})\}$$
$$\cdot\{30+(2x_1-3x_2)^2\cdot(18-32x_1+12x_1^{\,2}+48x_2-36x_1x_2+27x_2^{\,2})\}$$

Subject to

$$g_1(x_1,x_2)=-2-x_1\le 0,\ g_2(x_1,x_2)=-2-x_2\le 0,$$
$$g_3(x_1,x_2)=x_1-2\le 0,\ g_4(x_1,x_2)=x_2-2\le 0$$





A : Global Minimum

$x_1^* = 0.0, x_2^* = -1.0, f(x_1^*, x_2^*) = 3.0$

B : Local Minimum

$x_1^* = -0.6, x_2^* = -0.4, f(x_1^*, x_2^*) = 30.0$

C : Local Minimum

$x_1^* = 1.2, x_2^* = 0.8, f(x_1^*, x_2^*) = 840.0$

D : Local Minimum

$x_1^* = 1.8, x_2^* = 0.2, f(x_1^*, x_2^*) = 84.0$

6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예 - EzPreCompiler용 입력 파일 작성

```
$$ EzOptimizer Start
[Optimization Method]
    MFD
[Print Option]
    SMALL
[Design Variables]
    x1, 0.0, -2.0, 2.0
    x2, 0.0, -2.0, 2.0
[Objective Function]
    MINIMIZE f = (1.0+pow(x1+x2+1.0, 2.0)*(19.0-14.0*x1+3.0*pow(x1, 2.0)-
    14.0*x2+6.0*x1*x2+3.0 *pow(x2, 2.0))) * (30.0+pow(2.0*x1-3.0*x2, 2.0)*
    (18.0-32.0*x1+12.0*pow(x1, 2.0)+48.0*x2-36.0*x1*x2+27.0*pow(x2, 2.0)))
$$ EzOptimizer End
```

← 제약 조건을 설계 변수에 대한 상·하한값으로 표현

6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예 - EzPreCompiler에 의해 자동 생성된 C++ 파일

```
EzOptimizerConfiguration MyOptimizerConfiguration00(MFD, SMALL, 2, 0, MINIMIZE);
EzOptimizer MyOptimizer00(MyOptimizerConfiguration00);

int ResultFlag = 0;
double* Design_Value = MyOptimizer00.GetDesignValueAddress();
double* Lower_Value = MyOptimizer00.GetLowValueAddress();
double* Upper_Value = MyOptimizer00.GetUpperValueAddress();
double* Objective_Value = MyOptimizer00.GetObjectiveValueAddress();

Low_Value[ 0] = -2.000000; Design_Value[ 0] = 0.000000; Upper_Value[ 0] = 2.000000;
Low_Value[ 1] = -2.000000; Design_Value[ 1] = 0.000000; Upper_Value[ 1] = 2.000000;
```

➡ 제약 조건을 설계 변수에 대한 상·하한값으로 표현

```
while ((ResultFlag = MyOptimizer00.Optimization()) == 1) {
    *Objective_Value = (1.0+pow(Design_Value[0]+Design_Value[1]+1.0, 2.0)*(19.0-
    14.0*Design_Value[0]+3.0*pow(Design_Value[0], 2.0)-
    14.0*Design_Value[1]+6.0*Design_Value[0]*Design_Value[1] +3.0 *pow(Design_Value[1], 2.0))) *
    (30.0+pow(2.0*Design_Value0]-3.0*Design_Value[1], 2.0)* (18.0-
    32.0*Design_Value[0]+12.0*pow(Design_Value[0], 2.0)+48.0*Design_Value[1]-
    36.0*Design_Value[0]*Design_Value[1]+27.0*pow(Design_Value[1], 2.0)));
}

if (ResultFlag == 0) { x1 = Design_Value[ 0]; x2 = Design_Value[ 1]; f = *Objective_Value; }
else if (ResultFlag == -1) { MyOptimizer00.GetErrorMessage(); }
```

6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예 - EzOptimizer를 이용한 최적화 결과

최적화 알고리즘에 따른 결과 비교

	True Solution	MFD	MS	GA	HYBRID w/o Refine	HYBRID with Refine
x1	0.0000	-0.6000	0.0000	0.0081	0.0000	0.0000
x2	-1.0000	-0.4000	-1.0000	-1.0032	-1.0000	-1.0000
f	3.0000	30.0000	3.0000	3.0262	3.0000	3.0000
Iteration No	-	5	154	78	33	36
CPU Time(s)		0.03	0.78	1.23	0.73	0.75

Local Minimum ⬆

⬆

개선된 유전 알고리즘으로부터 얻어진 근사 최적점에 대해
local optimization**을 수행함으로써 정확한 전역 최적해 도출**

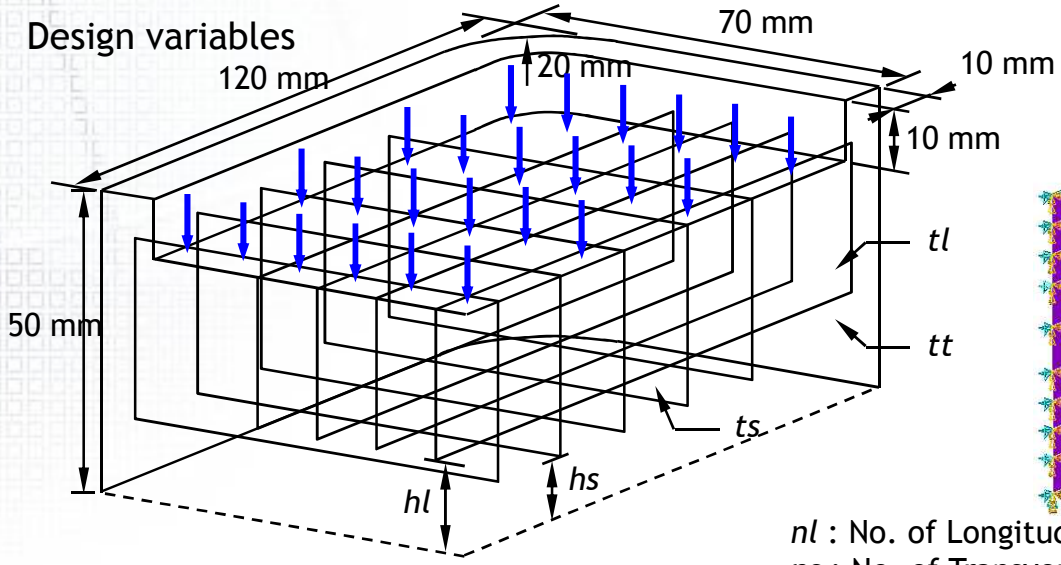
* MFD: Method of feasible directions, MS: Multi-start local optimization method, GA: Genetic algorithm, HYBRID: Global-local hybrid optimization method
* **테스트 시스템**: Pentium 3 866MHz, 512MB RAM

6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예 - Ribbed Tray 설계 문제

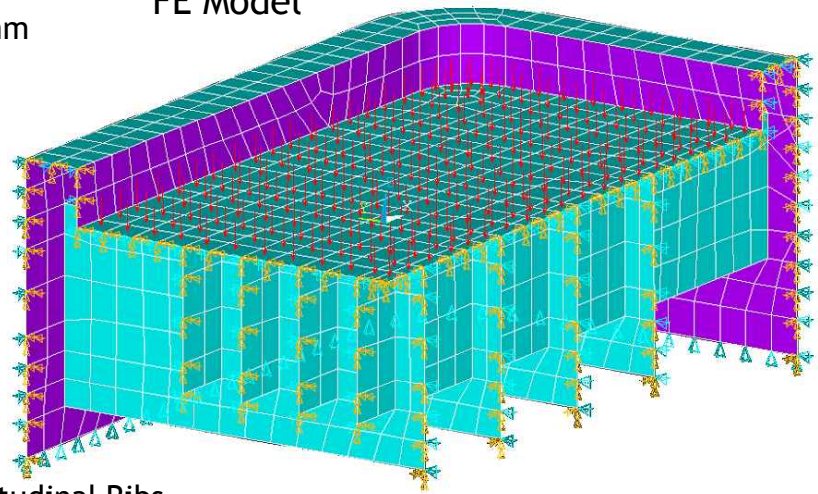
Minimize $Volume = f(nl, ns, tl, ts, tt, hl, hs)$ **Objective Function**

Subject to $\delta_{\max} \le 1.0(mm)$ $\sigma_{\max} \le 22.5(MPa)$ **Constraints**

Design variables



FE Model



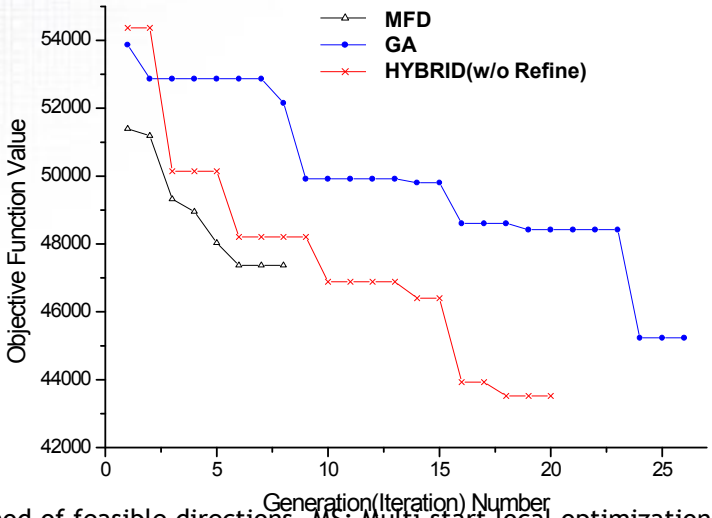
nl : No. of Longitudinal Ribs
ns : No. of Transversal Ribs
tl : Thickness of Longitudinal Ribs
ts : Thickness of Transversal Ribs
tt : Thickness of Tray
hl : Gap at Bottom of Longitudinal Ribs
hs : Gap at Bottom of Transversal Ribs

E = 3.1×10^3 *MPa*
ν = 0.35
Surface Load on Cavity = 0.15 *MPa*

* Kyu-Yeul Lee, Seon-Ho Cho, Myung-Il Roh, "An Efficient Global-Local Hybrid Optimization Method Using Design Sensitivity Analysis", International Journal of Vehicle Design(SCIE), Vol. 28, No. 4, pp.300~317, 2002.7

6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예 - Ribbed Tray 문제에 대한 최적화 결과(1)

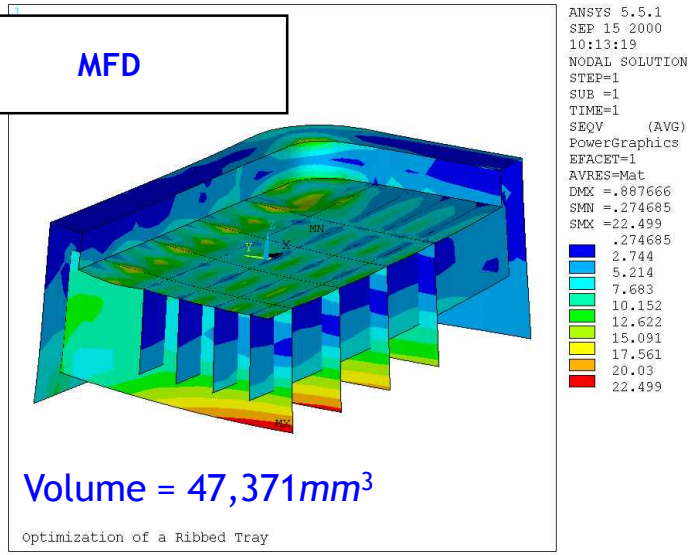
	Unit	MFD	GA	HYBRID	
				w/o Refine	with Refine
Volume	mm^3	47,370.662	45,236.305	43,520.642	42,507.595
nl		4	3	4	4
ns		4	4	4	4
tl	*mm*	1.307021	1.199413	1.007820	0.995354
ts	*mm*	1.313513	1.355816	1.332356	1.327935
tt	*mm*	1.067905	1.156403	1.025415	1.008237
hl	*mm*	16.779161	16.434995	16.998045	18.129324
hs	*mm*	6.073858	5.619746	5.482893	5.389428
Iteration No	-	8	26	20	25
CPU Time	*sec*	2,420.38	13,091.02	19,767.95	20,973.59



* MFD: Method of feasible directions, MS: Multi-start local optimization method, GA: Genetic algorithm, HYBRID: Global-local hybrid optimization method
* **테스트 시스템**: Pentium 3 866MHz, 512MB RAM

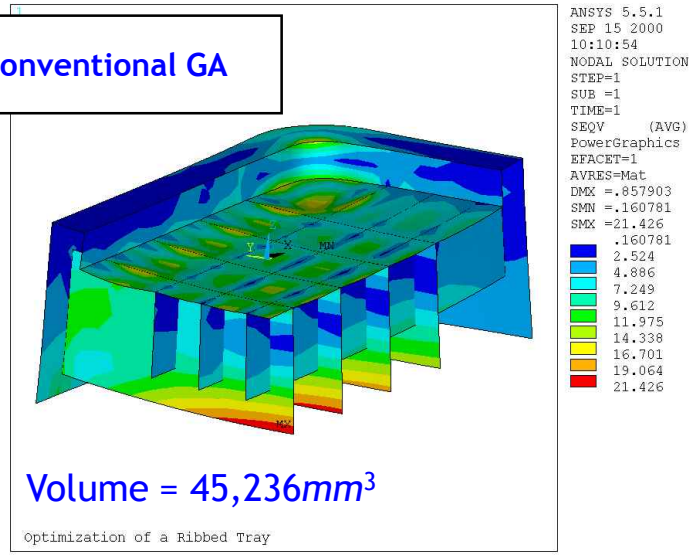
6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예 - Ribbed Tray 문제에 대한 최적화 결과(2)

MFD



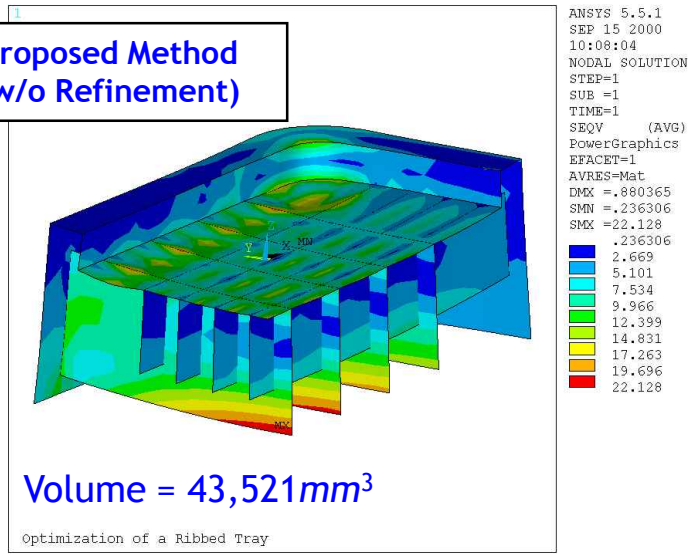
Volume = 47,371mm^3

Conventional GA



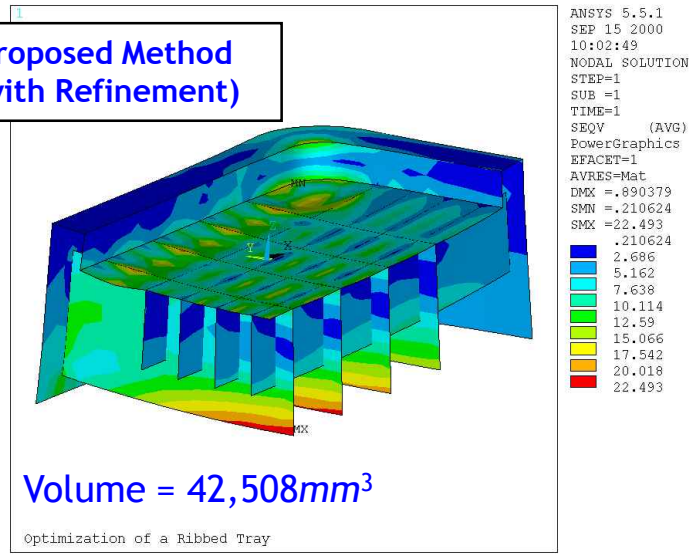
Volume = 45,236mm^3

Proposed Method (w/o Refinement)



Volume = 43,521mm^3

Proposed Method (with Refinement)



Volume = 42,508mm^3

6.3 제약 비선형 최적화 프로그램을 이용한 최적 설계 예 - 건조비를 최소화 하는 중앙 단면 설계

Find $x_i, i = 1, \cdots, 16$

Minimize *Building Cost*

Subject to

$$t_{i,\min} - x_i \le 0, \; i = 6, \cdots, 16$$

: minimum plate thickness

$$Z_{\min}^{deck} - Z^{deck} \le 0$$

: minimum section modulus at deck

$$Z_{\min}^{bottom} - Z^{bottom} \le 0$$

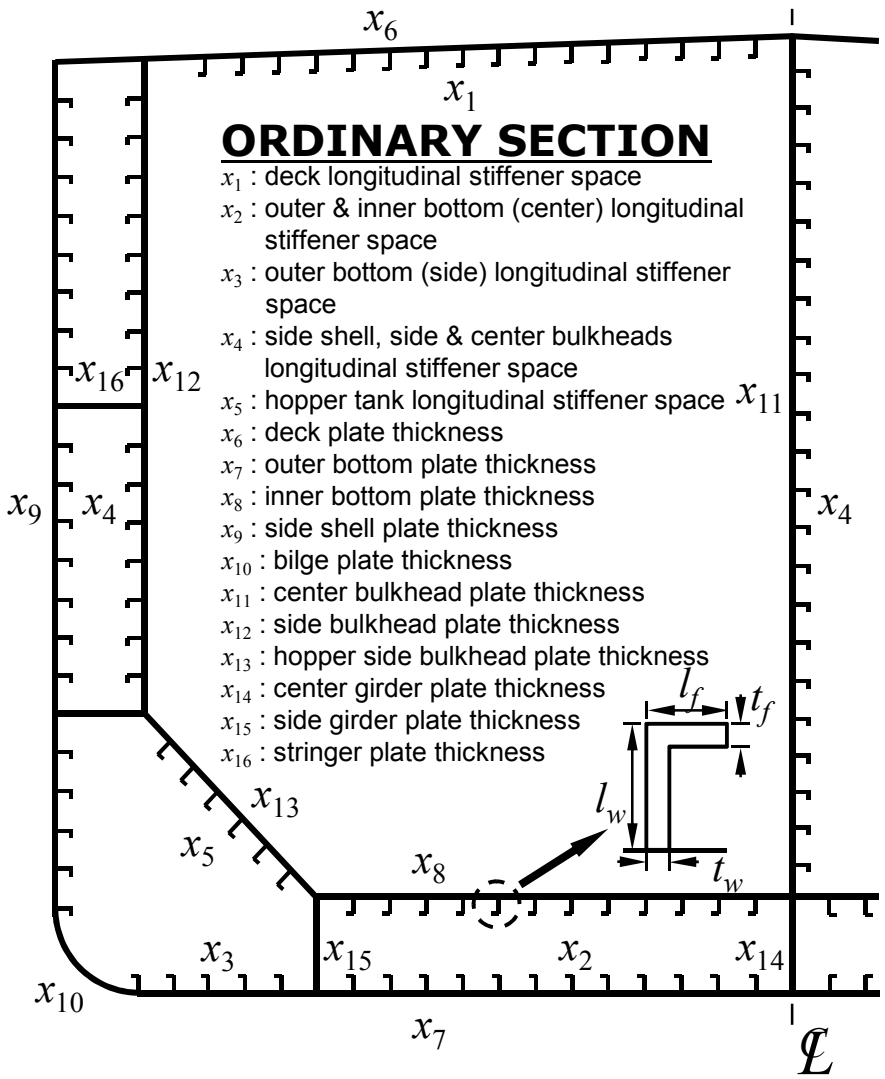
: minimum section modulus at bottom

$$\sigma^{deck} - \eta^{deck}\sigma_c^{deck} \le 0$$

: critical buckling stress at deck

$$\sigma^{bottom} - \eta^{bottom}\sigma_c^{bottom} \le 0$$

: critical buckling stress at bottom



* Kyu-Yeul Lee, Myung-Il Roh, "An Efficient Genetic Algorithm Using Gradient Information for Ship Structural Design Optimization", Journal of Ship Technology Research, Vol. 48, No. 4, pp.161-170, 2001.

건조비를 최소화 하는 중앙 단면 설계 문제의 수학적 정식화(1)

■ 목적 함수

- Midship Section의 단위길이당 건조비

$$Building\ Cost = Material\ Cost + Labour\ Cost \quad [\$/m]$$

$$Material\ Cost = Weight\,[ton] \times Unit\ Material\ Cost\,[\$/ton]$$

: midship section의 단위 길이당 재료비

$$LabourCost = Welding\ Cost + Painting\ Cost$$

: midship section의 단위 길이당 인건비

$$Weight = \rho \cdot A \quad [ton]$$

: midship section의 단위 길이당 중량

$$Welding\ Cost = 2 \cdot Fw \cdot N \quad [\$/m]$$

: midship section의 단위 길이당 용접비

$$Painting\ Cost = Fp \cdot G \quad [\$/m]$$

: midship section의 단위 길이당 도장비

ρ : steel's mass density, 7.85[ton/m^3]
A : area of the midship section
Fw: welding cost per unit length
N : number of the plates and stiffeners
Fp : painting cost per unit area
G : summation of the girth length of the plates and stiffeners

건조비를 최소화 하는 중앙 단면 설계 문제의 수학적 정식화(2)

■ 제약 조건

- Minimum Plate Thickness에 관한 조건

$$t_{i,\min} - x_i \le 0\,,\ \ i = 6, \cdots, 16$$

$$t_{i,\min} = \max\left(\frac{15.8 \cdot s_i \cdot \sqrt{p_i}}{\sqrt{\sigma_i}} + t_k, c_i + \frac{k_i \cdot L}{\sqrt{f_i}} + t_k \right) \ [mm]$$

- Minimum Section Modulus에 관한 조건(Deck, Bottom)

$$Z_{\min}^{deck} - Z^{deck} \le 0 \quad Z_{\min}^{bottom} - Z^{bottom} \le 0$$

$$Z_{\min}^{deck} = \max\left(\frac{C_W \cdot L^2 \cdot B \cdot (C_B + 0.7)}{1.39}, \frac{|M_S + M_W|}{175 \cdot 1.39} \right) \ [cm^3]$$

$$Z_{\min}^{bottom} = \max\left(\frac{C_W \cdot L^2 \cdot B \cdot (C_B + 0.7)}{1.28}, \frac{|M_S + M_W|}{175 \cdot 1.28} \right) \ [cm^3]$$

건조비를 최소화 하는 중앙 단면 설계 문제의 수학적 정식화(3)

- Critical Buckling Stress에 관한 조건(Deck, Bottom)

$$\sigma^{deck} - \eta^{deck}\sigma_c^{deck} \le 0 \qquad \sigma^{bottom} - \eta^{bottom}\sigma_c^{bottom} \le 0$$

$$\sigma_c^{deck} = \begin{cases} \sigma_{el}^{deck} & \text{when } \sigma_{el}^{deck} < 177.5 \\ 355 \cdot \left(1 - \dfrac{355 \cdot \sigma_{el}^{deck}}{4}\right) & \text{when } \sigma_{el}^{deck} > 177.5 \end{cases}$$

$$\sigma_c^{bottom} = \begin{cases} \sigma_{el}^{bottom} & \text{when } \sigma_{el}^{bottom} < 117.5 \\ 235 \cdot \left(1 - \dfrac{355 \cdot \sigma_{el}^{bottom}}{4}\right) & \text{when } \sigma_{el}^{bottom} > 117.5 \end{cases}$$

$$\sigma^{deck} = \frac{M_S + M_W}{I_N} \cdot z_n^{deck} \cdot 10^5 \quad [N/mm^2]$$

$$\sigma^{bottom} = \frac{M_S + M_W}{I_N} \cdot z_n^{bottom} \cdot 10^5 \quad [N/mm^2]$$

중앙 단면 설계 문제에 대한 최적화 결과

	Unit	Actual Ship	MFD	MS	GA	HYBRID	
						w/o Refine	with Refine
Building Cost	*$/m*	-	21,035.254748	20,637.828634	20,597.330090	20,422.478135	20,350.286893
x_1	*mm*	800.0	787.038274	811.324938	780.000000	810.000000	810.3701321
x_2	*mm*	800.0	762.891023	799.038243	750.000000	800.000000	800.1282732
x_3	*mm*	780.0	743.313979	787.034954	770.000000	790.000000	789.0923943
x_4	*mm*	835.0	814.142029	833.909455	820.000000	830.000000	834.838424
x_5	*mm*	770.0	756.434513	772.349435	790.000000	780.000000	780.002092
x_6	*mm*	16.5	16.983723	16.203495	16.000000	16.000000	16.390923
x_7	*mm*	16.0	16.829142	16.043803	16.500000	16.000000	15.989044
x_8	*mm*	15.5	16.020913	15.390394	16.000000	15.500000	15.432091
x_9	*mm*	17.0	17.329843	17.039439	16.500000	16.500000	17.139433
x_{10}	*mm*	14.5	15.001923	14.324335	15.000000	15.000000	14.780908
x_{11}	*mm*	13.5	14.192834	14.240495	14.000000	13.500000	13.550214
x_{12}	*mm*	14.5	15.123051	15.403945	14.500000	14.500000	14.500130
x_{13}	*mm*	17.0	16.902832	16.849387	16.500000	17.000000	17.010902
x_{14}	*mm*	14.0	14.784034	14.739454	15.500000	14.500000	14.309324
x_{15}	*mm*	14.0	15.129430	14.448504	15.500000	14.500000	14.588917
x_{16}	*mm*	14.5	14.824045	14.940584	15.000000	15.000000	14.789992
Iteration No	-	-	8	912	93	64	70
CPU Time	*sec*	-	2.90	293.28	272.91	265.06	267.92

* 최적 설계값을 실제로 사용하기 위해서는 부재의 두께와 간격의 경우 생산성을 고려하여 적절한 조정(예, 반올림 등)이 필요함