내부점 선형계획법의 밀집열 분할에 대하여*

설동렬** · 박순달** · 정호원***

On Dense Column Splitting in Interior Point Methods of Linear Programming*

Tongryeol Seol**, Soondal Park**, Howon Jung***

ABSTRACT

The computational speed of interior point method of linear programming depends on the speed of Cholesky factorization. If the coefficient matrix A has dense columns then the matrix $A\Theta A^T$ becomes a dense matrix. This causes Cholesky factorization to be slow. We study an efficient implementation method of the dense column splitting among dense column resolving techniques and analyze the relation between dense column splitting and ordering methods to improve the sparsity of Cholesky factor.

1. 서 론

모든 내부점 선형계획법은 대칭양정치 행렬의 선형방정식을 푸는 과정이 필요하다. 일반적으로 선형계획법 문제들은 희소한 특성을 가지는데, 내부점 선형계획법에서 나타나는 대칭양정치 행렬도 희소행렬이다[2]. 내부점 선형계획법에서는 전체 해법 소요 시간의 많은 부분이 선형방정식을 푸는 데에 소요되어, 효율적으로 선형방정식을 풀어내는 것이 해법의 성능을 크게 좌우한다.

내부점 선형계획법에서 나타나는 대칭양정치 행렬은 $A\Theta A^T$로 표현한다. 단, 행렬 A는 $m\times n$ 행렬이고, Θ는 $n\times n$의 대각행렬이고 모든 대각요소의 값은 양이다. Θ는 내부점 선형계획법의 종류에 따라서 다르게 계산된다. 대칭양정치 행렬 $A\Theta A^T$는 $m\times m$ 행렬이다. 내부점 선형계획법에서의 선형방정식은 다음과 같다.

$$A\Theta A^T\boldsymbol{x} = \boldsymbol{b}$$

단, $\boldsymbol{x}$과 $\boldsymbol{b}$는 $m\times 1$ 벡터이다.

행렬 A에 밀집열(dense column)이 존재하면 행렬 $A\Theta A^T$의 밀집도는 매우 높아지는 성질을

* 본 연구는 한국과학재단의 목적기초연구과제(과제번호 95-0200-39-01-2)에 의해 지원되었음
** 서울대학교 산업공학과
*** 고려대학교 경영학과

가지고 있다. Θ가 대각행렬이기 때문에 $A\Theta A^T$의 비영요소 구조에 영향을 주지 않는다. 따라서, $A\Theta A^T$의 비영요소 구조는 AA^T의 비영요소 구조와 동일하다. 행렬 AA^T의 (i,j)요소는 행렬 A의 i번째 행과 행렬 A^T의 j번째 열(즉, 행렬 A의 j번째 행)을 곱해서 얻는다. AA^T의 (i,j)요소가 비영요소이려면 행렬 A의 i번째 행과 j번째 행에서 모두 비영요소를 가지는 열이 적어도 하나 존재해야 한다. 거꾸로 이야기하면, 행렬 A의 k번째 열에서 비영요소를 가진 행들의 쌍들은 모두 AA^T에서 비영요소를 가지게 된다. 따라서, 행렬 A의 전체적인 밀집도는 낮다고 할지라도 특정한 열의 밀집도가 높은 경우에는 행렬 $A\Theta A^T$의 밀집도를 높이게 된다. 행렬 $A\Theta A^T$의 밀집도가 높으면 선형방정식을 푸는 데에 많은 시간이 소요된다. 이것은 내부점 선형계획법의 수행속도를 저하시키는 결과를 가져오게 된다.

행렬 A에 존재하는 밀집열을 해결하기 위한 기법으로, Schur complement를 이용하여 밀집열를 제외한 나머지 부분에 대해서만 촐레스키 분해를 적용하는 Partitioning 방법[4]이 있다. Schur complement 방법은 밀집열을 따로 처리하므로 촐레스키 분해 속도는 매우 빠르게 개선되지만, 밀집열을 제외한 나머지 부분이 *rank*가 아닐 수 있기 때문에 수치적으로 안정적이지 않은 문제점이 있다.

한편, 촐레스키 분해를 사용하지 않고 반복적인 방법을 사용하여 선형방정식을 풀 수 있다. 예를 들어, Conjugate gradient 방법[1]이 네트워크 문제와 같은 특수한 형태의 행렬을 가지는 문제에 대해 내부점 방법을 적용할 때에 주로 사용된다.

쌍대문제 전환 기법은 Vanderbei[7]에 의해 제안되었는데, 입력된 문제를 쌍대문제로 전환하여 밀집열을 제거하는 방법이다. 쌍대문제 전환 기법은 밀집행이 존재하여 쌍대문제에서도 밀집열이 있는 경우에는 효과가 없는 방법이다.

밀집열분할(dense column splitting) 방법은 Vanderbei[8]에 의해서 제안된 방법으로 밀집열을 다루기에 효율적인 방법으로 알려져 있다.[3] Gondzio[5]는 밀집열을 분할할 때에 행구조를 이용하여 희소도를 향상시키 위한 알고리즘을 제시하였다. 밀집열 처리를 위해서 사용되는 Schur complement 방법에서는 새로 생성되는 행렬의 *rank*가 *full rank*가 아닐 수 있기 때문에 수치적 안정성이 문제가 되지만, 밀집열분할 방법을 사용할 경우에는 밀집열 분할에 의해 변형된 행렬에 대해서 *full rank*가 보장되므로 수치적으로 안정적이다.[5] 그러나, 밀집열의 개수가 많을 경우에는 분할방법이 효과적이지 않은 것으로 보고되어 있다.

본 논문에서는 선형계획법 문제에 밀집열 분할 방법을 적용하기 위한 이론적 근거를 제시하였다. 선형계획법 문제에 밀집열 분할 방법을 적용하면 새로운 변수와 제약식이 원래의 문제에 추가되어 원래 문제가 변형된다. 본 논문에서는 밀집열 분할에 의해서 변형된 새로운 선형계획법 문제에 대해서 구한 최적해로부터 원래의 선형계획법 문제의 최적해를 쉽게 구할 수 있음을 보였다.

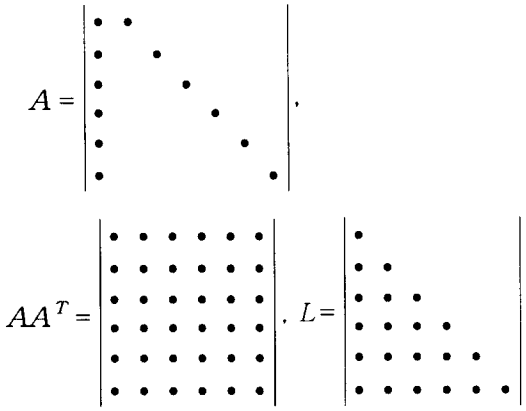
밀집열 분할을 적용할 때에 희소성을 보다 향상시키기 위하여 두 가지 접근 방법에 대하여 고려하였다. 첫째는 밀집열 분할을 수행할 때에 AA^T를 희소행렬이 되도록 하는 것이고, 둘째는 밀집열 분할 방법을 적용한 다음 촐레스키 분해를 적용하여 얻는 분해행렬 L이 희소성을 유지하도록 하는 것이다.

첫째 접근 방법에서는 여러 개의 밀집열이 존재할 때에 밀집열에 속한 비영요소의 행지수들을 분할된 열에 적절히 배정함으로써 AA^T의 비영

요소수를 줄일 수 있음을 보였다. 둘째 접근 방법에서는 순서화 방법을 고려하였다. Vanderbei나 Gondzio는 밀집열이 많은 경우에는 밀집열 분할이 적절하지 않다고 하였다. 밀집열 분할 후에 최소 차수 순서화를 적용하면 AA^T의 비영요소수는 작지만 촐레스키 분해행렬 L의 비영요소수가 매우 증가하는 경우가 있기 때문이다. 본 논문에서는 밀집열 분할에 의한 AA^T의 비영요소구조가 가지는 특성을 분석하여 적합한 순서화 방법에 대하여 연구하였다.

2. 하나의 밀집열을 분할하는 방법

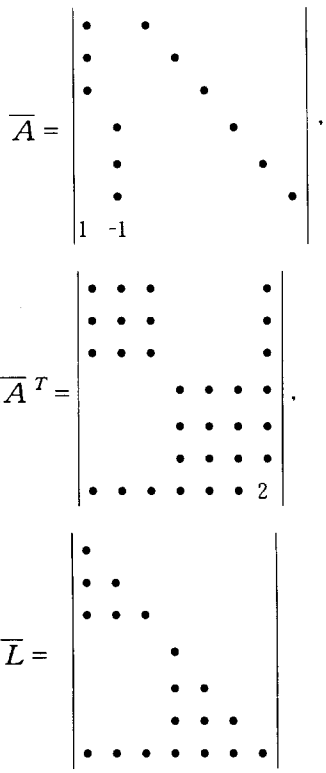
밀집열분할 방법의 기본 개념을 설명하기 위해서 [그림 1]과 같은 밀집열을 가진 6×7 행렬 A를 고려하자.



[그림 1] 밀집열을 가진 행렬의 형태

첫 번째 열이 밀집열이기 때문에 전체적으로 희소행렬이지만, 내부점 선형계획법을 수행할 때에 계산되는 AA^T행렬은 밀집행렬로 나타나고, 결과적으로 촐레스키 분해행렬 L도 역시 밀집행렬이 된다. 따라서, 원래 행렬 A가 가졌던 희소행렬의 좋은 특성을 전혀 이용할 수 없게 된다. 한편, [그림 2]의 행렬 $\overline{A}$는 행렬 A의 첫 번째 열을 두 개의 열로 분할한 행렬이다.

$\boldsymbol{x}=(x_1,x_2,x_3,x_4,x_5,x_6)^T$에서 변수 x_1를 x_1, x_1'의 두 개 변수로 쪼개고 마지막에 $x_1-x_1'=0$ 제약식을 추가한다.



[그림 2] 밀집열이 분할된 행렬의 형태

행렬 $\overline{L}$은 L행렬 보다 차수는 증가하였지만, 희소한 성질이 유지되었음을 볼 수 있다. 일반적인 행렬에 대해서 밀집열분할은 다음과 같이 적용된다. $m\times n$행렬 A와 이에 따른 벡터 $\boldsymbol{b}$, $\boldsymbol{c}$, $\boldsymbol{x}$를 고려하자.

$$\text{(P1)}\quad \begin{aligned} &\min\ \boldsymbol{c}^T\boldsymbol{x}\\ &s.t.\ A\boldsymbol{x}=\boldsymbol{b}\\ &\quad\ \boldsymbol{x}\geq\boldsymbol{0}\end{aligned}$$

단, $\boldsymbol{b}=(b_1, b_2, \cdots, b_m)^T$,

$\boldsymbol{c}=(c_1, c_2, \cdots, c_n)^T$,

$\boldsymbol{x}=(x_1, x_2, \cdots, x_n)^T$

행렬 A의 i번째 열을 $\boldsymbol{a}_i$라고 하자. $\boldsymbol{a}_1$은 밀집열이고 p개의 요소가 비영요소라고 가정한다. $\boldsymbol{a}_1$의 q개의 비영요소를 가지는 열을 $\bar{\boldsymbol{a}}_1$라고 하고, 나머지 $p-q$개의 비영요소를 가지는 열을 $\bar{\boldsymbol{a}}_2$라고 하자. 그리고, $\bar{\boldsymbol{a}}_1$과 $\bar{\boldsymbol{a}}_2$의 비영요소만을 가지는 벡터를 각각 $\boldsymbol{a}_1^1$, $\boldsymbol{a}_1^2$라고 하자. 즉,

$$\boldsymbol{a}_1=\begin{pmatrix} a_{11} \\ \vdots \\ a_{q1} \\ a_{q+1,1} \\ \vdots \\ a_{p1} \end{pmatrix},\quad \bar{\boldsymbol{a}}_1=\begin{pmatrix} a_{11} \\ \vdots \\ a_{q1} \\ 0 \\ \vdots \\ 0 \end{pmatrix},\quad \bar{\boldsymbol{a}}_2=\begin{pmatrix} 0 \\ \vdots \\ 0 \\ a_{q+1,1} \\ \vdots \\ a_{p1} \end{pmatrix},$$

$$\boldsymbol{a}_1^1=\begin{pmatrix} a_{11} \\ \vdots \\ a_{q1} \end{pmatrix},\quad \boldsymbol{a}_1^2=\begin{pmatrix} a_{q+1,1} \\ \vdots \\ a_{p1} \end{pmatrix}$$

$\boldsymbol{a}_1$의 첫 p개의 요소가 비영요소라고 하면 $\boldsymbol{a}_1x_1=\bar{\boldsymbol{a}}_1x_1+\bar{\boldsymbol{a}}_2x_1$이 되도록 두 개의 열로 분할하면 다음과 같다. (P1)의 제약식을 만족시키면서 변수 x_1을 두 개의 변수 x_1과 x_1'으로 나누기 위해서 $x_1-x_1'=0$를 제약식에 추가하면 $\boldsymbol{a}_1x_1$은 결과적으로 다음과 같은 두 개의 열로 분할된다.

$$\begin{pmatrix} a_{11} \\ \vdots \\ a_{q1} \\ a_{q+1,1} \\ \vdots \\ a_{p1} \end{pmatrix}x_1=\begin{pmatrix} a_{11} \\ \vdots \\ a_{q1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}x_1+\begin{pmatrix} 0 \\ \vdots \\ 0 \\ a_{q+1,1} \\ \vdots \\ a_{p1} \end{pmatrix}x_1$$

$$\Rightarrow \quad \begin{pmatrix} a_{11} \\ \vdots \\ a_{q1} \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}x_1+\begin{pmatrix} 0 \\ \vdots \\ 0 \\ a_{q+1,1} \\ \vdots \\ a_{p1} \\ -1 \end{pmatrix}x_1'$$

행렬 A에서 $\boldsymbol{a}_1$를 제외한 나머지 열들로 이루어진 행렬을 A_0라고 하자. 그러면, 행렬 A에서 $\boldsymbol{a}_1$을 위와 같이 두 개의 열로 분할한 행렬 은 $\bar{A}$, $\bar{\boldsymbol{b}}$, $\bar{\boldsymbol{c}}$, $\bar{\boldsymbol{x}}$ 다음과 같다.

$$\bar{A}=\begin{pmatrix} \bar{\boldsymbol{a}}_1 & \bar{\boldsymbol{a}}_2 & A_0 \\ 1 & -1 & 0 \end{pmatrix},$$

$$\bar{\boldsymbol{b}}=(b_1, b_2, \cdots, b_m, 0)^T,$$

$$\bar{\boldsymbol{c}}=(c_1, 0, c_2, \cdots, c_n)^T,$$

$$\bar{\boldsymbol{x}}=(x_1, x_1', x_2, \cdots, x_n)^T$$

그리고, 다음의 새로운 선형계획법 문제를 얻는다.

$$\text{(P2)} \qquad \begin{aligned} \min\ & \bar{\boldsymbol{c}}^T\bar{\boldsymbol{x}} \\ s.t.\ & \bar{A}\bar{\boldsymbol{x}}=\bar{\boldsymbol{b}} \\ & \bar{\boldsymbol{x}}\geq \boldsymbol{0} \end{aligned}$$

정리 1. (P2)의 가능해 $\bar{\boldsymbol{x}}=(\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_{n+1})$에 대해서, $\boldsymbol{x}=(x_1, x_2, \cdots, x_n)^T$가 (P1)의 가능해일 필요충분조건은

$$\begin{aligned} &\bar{x}_1=x_1, \\ &\bar{x}_2=x_1, \\ &\bar{x}_i=x_{i-1},\quad i=3, 4, \cdots, n+1 \end{aligned}$$

이다.

(증명)

($\Rightarrow$)

(P1)의 가능해 $\boldsymbol{x}=(x_1, x_2, \cdots, x_n)^T$에 대하여 $x_1' = x_1$인 $\bar{\boldsymbol{x}} = (x_1, x_1, x_2, \cdots, x_n)^T$는 (P2)의 가능해임은 자명하다.

(⇐)

$\bar{\boldsymbol{x}}$가 (P2)의 가능해이므로, $\bar{x}_1 - \bar{x}_1' = 0$이다. $\bar{x}_1' = \bar{x}_1$으로 대입하면, $\bar{A}\bar{\boldsymbol{x}}$의 첫 두 열은 다음과 같이 된다.

$$\begin{aligned}\begin{pmatrix}\bar{\boldsymbol{a}}_1\\1\end{pmatrix}\bar{x}_1 + \begin{pmatrix}\bar{\boldsymbol{a}}_2\\-1\end{pmatrix}\bar{x}_1' &= \begin{pmatrix}\bar{\boldsymbol{a}}_1\\1\end{pmatrix}\bar{x}_1 + \begin{pmatrix}\bar{\boldsymbol{a}}_2\\-1\end{pmatrix}\bar{x}_1\\ &= \begin{pmatrix}\bar{\boldsymbol{a}}_1 + \bar{\boldsymbol{a}}_2\\0\end{pmatrix}\bar{x}_1\\ &= \begin{pmatrix}\boldsymbol{a}_1\\0\end{pmatrix}\bar{x}_1\end{aligned}$$

즉,

$$\begin{aligned}\bar{A}\bar{\boldsymbol{x}} &= \begin{pmatrix}\bar{\boldsymbol{a}}_1 & \bar{\boldsymbol{a}}_2 & A_0\\1 & -1 & 0\end{pmatrix}\bar{\boldsymbol{x}}\\ &= \begin{pmatrix}\boldsymbol{a}_1 & A_0\\0 & 0\end{pmatrix}\boldsymbol{x}\\ &= \bar{\boldsymbol{b}} = \begin{pmatrix}\boldsymbol{b}\\0\end{pmatrix}\end{aligned}$$

이고 $A = (\boldsymbol{a}_1\ A_0)$이므로, $A\boldsymbol{x} = \boldsymbol{b}$이다. 따라서, $\boldsymbol{x}$는 (P1)의 가능해이다.

■

정리 2. (P2)의 최적해

$$\bar{\boldsymbol{x}}^* = (\bar{x}_1^*, \bar{x}_1'^*, \bar{x}_2^*, \cdots, \bar{x}_n^*)^T$$

에 대해서 (P1)의 최적해는

$$\boldsymbol{x}^* = (x_1^*, x_2^*, \cdots, x_n^*)^T = (\bar{x}_1^*, \bar{x}_2^*, \cdots, \bar{x}_n^*)^T$$

이다.

(증명)

정리 1에 의하여 (P2)의 최적해 $\bar{\boldsymbol{x}}^*$에 대해서 $\hat{\boldsymbol{x}} = (\bar{x}_1^*, \bar{x}_2^*, \cdots, \bar{x}_n^*)^T$는 (P1)의 가능해이다.

이제 $(\bar{x}_1^*, \bar{x}_2^*, \cdots, \bar{x}_n^*)^T$가 (P1)의 최적해임을 보이면 된다. $\bar{\boldsymbol{x}}^*$는 (P2)의 최적해이므로 (P2)의 모든 가능해 $\bar{\boldsymbol{x}}$에 대해서 다음을 만족한다.

$$\bar{\boldsymbol{c}}^T\bar{\boldsymbol{x}}^* \le \bar{\boldsymbol{c}}^T\bar{\boldsymbol{x}}$$

$\bar{\boldsymbol{c}} = (c_1, 0, c_2, \cdots, c_n)^T$이므로 $\boldsymbol{c}^T\boldsymbol{x} = \bar{\boldsymbol{c}}^T\bar{\boldsymbol{x}}$이다. 그러므로, 정리 1에 의하여 (P1)의 모든 가능해 $\boldsymbol{x}$에 대하여 다음을 만족한다.

$$\boldsymbol{c}^T\hat{\boldsymbol{x}} \le \boldsymbol{c}^T\boldsymbol{x}$$

따라서, $\hat{\boldsymbol{x}} = (\bar{x}_1^*, \bar{x}_2^*, \cdots, \bar{x}_n^*)^T$는 (P1)의 최적해이다.

■

정리 2에 의해서 (P1)에서 밀집열을 분할하여 얻은 새로운 문제 (P2)를 풀면, (P1)의 최적해를 바로 얻을 수 있고, (P2)의 최적목적함수값은 (P1)의 최적목적함수값과 같음을 알 수 있다.

(P1)과 (P2)의 쌍대문제 (D1)과 (D2)는 각각 다음과 같다.

$$\begin{aligned}&\max\ \boldsymbol{b}^T\boldsymbol{y}\\ (\mathrm{D1})\quad &s.t.\ A^T\boldsymbol{y} + \boldsymbol{z} = \boldsymbol{c}\\ &\boldsymbol{z} \ge \boldsymbol{0}\end{aligned}$$

단, $\boldsymbol{y} = (y_1, y_2, \cdots, y_m)^T$,

$\boldsymbol{z} = (z_1, z_2, \cdots, z_n)^T$

$$\begin{aligned}&\max\ \bar{\boldsymbol{b}}^T\bar{\boldsymbol{y}}\\ (\mathrm{D2})\quad &s.t.\ \bar{A}^T\bar{\boldsymbol{y}} + \bar{\boldsymbol{z}} = \bar{\boldsymbol{c}}\\ &\bar{\boldsymbol{z}} \ge \boldsymbol{0}\end{aligned}$$

단, $\bar{\boldsymbol{y}} = (y_1, y_2, \cdots, y_m, y_{m+1})^T$,

$$\bar{z}=(z_1, z_1', z_2, \cdots, z_n)^T$$

정리 3. $\bar{y}=(y_1, y_2, \cdots, y_m, y_{m+1})^T$ $\bar{z}=$, $(z_1, z_1', z_2, \cdots, z_n)^T$가 (D2)의 가능해일 필요충분조건은 $y=(y_1, y_2, \cdots, y_m)^T$, $z=(z_1+z_1', z_2, \cdots, z_n)^T$가 (D1)의 가능해인 것이다.

(증명)

(⇒)

(D2)의 가능해 $\bar{y}, \bar{z}$가 (D2)의 분할되지 않은 나머지 제약식을 만족하므로, $y=(y_1, y_2, \cdots, y_m)^T$, $z=(z_1, z_2, \cdots, z_n)^T$는 (D1)의 밀집행을 제외한 나머지 제약식을 만족시킨다.

한편, $\bar{y}$와 $\bar{z}$는 다음 식을 만족한다.

$$\begin{aligned} \bar{a}_1^T y + y_{m+1} + z_1 &= c_1 \\ \bar{a}_2^T y - y_{m+1} + z_1' &= 0 \end{aligned}$$

위의 두 식에서 y_{m+1}을 소거하면 다음의 식을 얻는다.

$$\begin{aligned} & \bar{a}_1^T y + \bar{a}_2^T y + z_1 + z_1' \\ = \; & a_1^T y + z_1 + z_1' \\ = \; & c_1 \end{aligned}$$

따라서, (D1)의 밀집행에 대해서도 만족한다.

(⇐)

$\bar{A}^T$의 분할되지 않은 행들에 대해서는 자명하므로, 분할된 행을 고려하자. y, z가 (D1)의 가능해이므로 다음을 만족한다.

$$a_1^T y + z_1 + z_1' = c_1$$

그리고, $y_{m+1}=\bar{a}_2^T y + z_1'$일 때,

$$\begin{aligned} c_1 &= a_1^T y + z_1 + z_1' \\ &= \bar{a}_1^T y + \bar{a}_2^T y + z_1 + z_1' \\ &= \bar{a}_1^T y + z_1 + y_{m+1} \end{aligned}$$

따라서, $\bar{y}=(y_1, y_2, \cdots, y_m, y_{m+1})^T$, $\bar{z}=(z_1, z_1', z_2, \cdots, z_n)^T$는 (D2)의 가능해이다.

■

정리 4. (D2)의 최적해

$$\bar{y}^*=(\bar{y}_1^*, \bar{y}_2^*, \cdots, \bar{y}_m^*, \bar{y}_{m+1}^*)^T,$$
$$\bar{z}^*=(\bar{z}_1^*, \bar{z}_1'^*, \bar{z}_2^*, \cdots, \bar{z}_n^*)^T$$

에 대해서 (D1)의 최적해는

$$\bar{y}^*=(\bar{y}_1^*, \bar{y}_2^*, \cdots, \bar{y}_m^*)^T,$$
$$\bar{z}^*=(\bar{z}_1^* + \bar{z}_1'^*, \bar{z}_2^*, \cdots, \bar{z}_n^*)^T$$

이다.

(증명)

정리 3에 의하여 (D2)의 최적해 $\bar{y}^*, \bar{z}^*$에 대해서

$$\hat{y}=(\bar{y}_1^*, \bar{y}_2^*, \cdots, \bar{y}_m^*)^T,$$
$$\hat{z}=(\bar{z}_1^* + \bar{z}_1'^*, \bar{z}_2^*, \cdots, \bar{z}_n^*)^T$$

는 (D1)의 가능해이다.

$\bar{y}^*, \bar{z}^*$는 (D2)의 최적해이므로 (D2)의 모든 가능해 $\bar{y}, \bar{z}$에 대해서 다음을 만족한다.

$$\bar{b}^T \bar{y}^* \geq \bar{b}^T \bar{y}$$

그런데, $\bar{b}=(b_1, b_2, \cdots, b_m, 0)^T$ 이므로

$b^T y = \bar{b}^T \bar{y}$이다. 그러므로 정리 3에 의하여 (D1)의 모든 가능해 y, z에 대하여 다음을 만족한다.

$$b^T \hat{y} \geq b^T y$$

따라서, $\hat{y}, \hat{z}$는 (D1)의 최적해이다.

■

정리 4에 의해서 밀집열이 분할된 문제 (D2)를 풀어서 얻은 쌍대최적해 $\bar{y}^*, \bar{z}^*$로부터 얻어지는 $\hat{y}, \hat{z}$가 원래의 문제 (D1)에 대해서 쌍대최적해가 됨을 알 수 있다.

3. 밀집열분할과 행렬의 희소도

Netlib에 속한 선형계획법 문제 가운데에 비영요소 개수가 50개 이상인 밀집열을 가진 문제는 [표 1]에서 보듯이 israel, seba, fit1p, fit2p 등이 있다. [표 1]에 있는 밀집열 문제들의 특성을 보면, 밀집열를 가지는 행렬에서 밀집열의 개수는 전체 열 개수에 비해서 매우 작지만, israel이나 seba는 한 열의 78%, 44%가 비영요소인 밀집열이 존재하고, fit1p와 fit2p는 완전밀집열(full column)을 가지고 있다. 따라서, 이들 행렬에서 A행렬은 희소행렬이지만, AA^T는 밀집도는 매우 높은 행렬이 된다. 특히, 완전밀집열을 가지는 fit1p나 fit2p의 경우에는 AA^T와 촐레스키 분해행렬 L이 완전밀집행렬(full matrix)을 이룬다.

밀집열분할은 촐레스키 분해행렬 L이 희소한 성질을 가지도록 하는 데에 목적이 있다. 밀집열 분할을 적용했을 때에 얻어지는 분해행렬 L의 희소도는 다음의 두 가지에 영향을 받는다.

- 분할방법
- 순서화 방법

밀집열을 분할하는 방법에 따라서 행렬 AA^T의 희소도가 좌우된다. 분해행렬 L의 비영요소 구조는 행렬 AA^T의 비영요소 구조를 포함하기 때문에 행렬 AA^T의 희소도를 낮게 유지하는 것이 일차적으로 중요하다. 한편, 밀집열분할을 사용할 때에는 순서화 방법에 따라서 행렬 AA^T를 분해하여 얻어지는 분해행렬 L의 밀집도가 크게 영향을 받기 때문에 적절한 순서화 방법을 적용하는 것이 필요하다.

분할방법과 AA^T의 희소도

여러 개의 밀집열을 분할하는 것은 한 개의 밀집열을 분할하는 방법을 각각의 밀집열에 대해서

[표 1] 밀집열문제의 특성

문제이름	행갯수	열갯수	비영요소수	밀집열수	밀집열에 속한 행수	밀집열의 최대 크기
israel	174	142	2,269	6	149	136
seba	515	1,028	4,352	14	341	230
fit1p	627	1,677	9,868	24	627	627
fit2p	3,000	13,525	50,284	25	3,000	3,000

$$\overline{AA}^T = \begin{pmatrix} a_1^1(a_1^1)^T + A_0^1(A_0^1)^T & A_0^1(A_0^2)^T & a_1^1 \\ A_0^2(A_0^1)^T & a_1^2(a_1^2)^T + A_0^2(A_0^2)^T & -a_1^2 \\ (a_1^1)^T & -(a_1^2)^T & 2 \end{pmatrix}$$

[그림 3] $\overline{AA}^T$의 구조

반복적으로 적용하면 된다. 따라서, 밀집열 분할을 k회 적용한다면 변수와 제약식의 개수가 각각 k개씩 증가한다.

여러 개의 밀집열을 분할할 때에 어떻게 밀집열을 분할하는가에 따라 결과적으로 얻어지는 $\overline{AA}^T$ 행렬의 희소도는 다양하게 나타날 수 있다. 행렬 A에서 밀집열 a_1를 두 개의 열로 분할한 $\overline{A}$가 다음과 같다고 하자. a_1^1과 a_1^2는 a_1를 2개로 분할한 벡터이고, A_0^1과 A_0^2는 분할되지 않은 나머지 열들로 이루어진 행렬을 분할된 행번호에 따라 나눈 것이다.

$$A = \begin{pmatrix} a_1^1 & A_0^1 \\ a_1^2 & A_0^2 \end{pmatrix} \Rightarrow \overline{A} = \begin{pmatrix} a_1^1 & 0 & A_0^1 \\ 0 & a_1^2 & A_0^2 \\ 1 & -1 & 0 \end{pmatrix}$$

$\overline{AA}^T$는 [그림 3]과 같은 구조를 가진다. a_1^1과 a_1^2는 밀집벡터이기 때문에 $a_1^1(a_1^1)^T$와 $a_1^2(a_1^2)^T$는 밀집행렬을 이룬다. 따라서, 행렬 $\overline{AA}^T$에서 $a_1^1(a_1^1)^T + A_0^1(A_0^1)^T$과 $a_1^2(a_1^2)^T + A_0^2(A_0^2)^T$은 밀집블럭이 되므로, $A_0^2(A_0^1)^T$와 $A_0^1(A_0^2)^T$ 부분을 제외한 나머지 부분의 비영요소수는 분할된 열의 비영요소수가 일정하면 분할방법에 상관없이 결정된다. 그러나, $A_0^2(A_0^1)^T$와 $A_0^1(A_0^2)^T$는 비영요소를 가지는 열번호가 같은 경우가 존재하는 행들이 A_0^1과 A_0^2에 골고루 분포되어 있을 경우에는 밀집도가 높아지고, 반대로 A_0^1과 A_0^2의 어느 한 쪽에만 모여 있을 경우에는 희소도가 높아진다. 이러한 현상은 밀집열이 여러 개일 경우에 각 밀집열에 대해서 비영요소 행이 어떻게 분포되도록 분할되었는가에 따라서 $\overline{AA}^T$의 밀집도에 큰 영향을 준다.[6] 따라서, $\overline{AA}^T$의 밀집도를 최소화하기 위해서는 $A_0^2(A_0^1)^T$와 $A_0^1(A_0^2)^T$의 비영요소수를 최소화하는 밀집열 분할 방법이 필요하다.

본 연구에서는 다음과 같은 분할 방법을 고려하였다. 즉, $A_0^2(A_0^1)^T$와 $A_0^1(A_0^2)^T$의 비영요소수를 줄이기 위해서 분할된 열들이 같은 행번호를 가지지 못하도록 한다. 밀집열에 속한 모든 열들의 행번호의 합집합을 구하여 행번호들의 합집합을 여러 개의 집합으로 먼저 분할한다. 각 밀집열에 대해서, 앞서 분할해 놓은 집합에서 같은 집합에 속한 열들만을 하나의 열로 분할한다.

[표 2]에서 분할방법 1은 행번호에 대하여 고려하지 않고 밀집열마다 독립적으로 분할을 수행

[표 2] 행렬 AA^T의 비영요소수

문제이름	분할 않음	분할방법 1	분할방법 2
israel	10,800	6,457	5,956
seba	51,274	15,773	12,044
fit1p	196,251	52,787	28,043
fit2p	4,498,500	254,029	141,802

[표 3] 행렬 L의 비영요소수

문제 이름	분할 않음	최소차수순서화	최소부족수순서화	수정된 최소차수순서화
israel	11,215	9,311	8,985	10,520
seba	56,837	18,094	16,766	28,776
fit1p	196,251	228,477	64,926	107,482
fit2p	4,498,500	*	390,614	1,474,103

* : 메모리 소요량이 많아서 계산을 수행하지 못했음

한 경우이다. 분할방법 2는 분할방법 1에 기초하여 전체 행번호 집합을 분할하는 방법을 적용한 것이다. 실험결과에 의하면 분할방법 2가 효과적인 것으로 나타났다. 즉, 분할방법 2가 $A_0^2(A_0^1)^T$와 $A_0^1(A_0^2)^T$의 비영요소수를 줄여주는 좋은 발견적 방법임을 알 수 있다.

순서화 방법과 L의 밀집도

밀집열분할 방법을 적용하면, 열을 분할할 때에 추가되는 행이 가지는 특성 때문에 행렬 AA^T은 밀집도는 줄어들지만, 최소 차수 순서화를 적용할 경우에 행렬 L의 밀집도가 매우 높아지는 현상이 발생할 수 있다. 추가된 행의 차수는 분할된 열의 비영요소수가 되지만, 이 점이 삭제되면 분할된 두 개의 열이 하나로 합쳐져서 밀집도가 높아지기 때문에 부족수는 상대적으로 큰 값을 가진다. 따라서, 차수가 같은 점들이 있을 때에 분할에 의해서 선택된 행이 먼저 삭제되면 행렬 L의 밀집도가 매우 높아지게 된다. 행렬 L의 밀집도를 개선하기 위해서는 최소 차수 순서화 대신 다음의 두 가지 방법을 적용할 수 있다.

- 최소 부족수 순서화

추가된 행의 부족수를 직접 사용하면, 추가된 행이 미리 삭제되는 것을 막을 수 있으므로 최소 부족수 순서화를 사용한다. 이 방법을 적용하면 행렬 L의 희소도는 좋아지지만, 최소 부족수 순서화의 수행시간이 최소 차수 순서화에 비해서 너무 오래 걸린다는 단점이 있다.

- 수정된 최소 차수 순서화

추가된 행이 순서화 도중에 미리 삭제되는 것을 막기 위해서, 추가된 행을 제외한 나머지 행에 대해서만 최소 차수 순서화를 적용하고, 추가된 행에 대해서는 최소 차수 순서화에 의해 결정된 순서에 이어서 차례로 순서를 매겨 준다. 최소 차수 순서화의 빠른 계산을 이용할 수 있고, 추가된 행에 의해서 행렬 L의 희소도가 깨어지는 것을 막을 수 있다.

[표 3]은 최소 차수 순서화, 최소 부족수 순서화 그리고 수정된 최소 차수 순서화를 각각 적용하였을 경우에 촐레스키 분해행렬 L의 비영요소수이다. israel과 seba에서는 최소 차수 순서화를 적용했을 때에 밀집열분할을 적용하지 않았을 때보다 비영요소수가 감소하였지만, fit1p와 같은 경우에는 오히려 밀집열분할을 적용하지 않았을 때보다 비영요소수가 증가하였다. 최소 부족수 순서화를 적용하였을 때에 비영요소수는 가장 적었고, 수정된 최소 차수 순서화를 적용하였을 때에는 밀집열분할을 적용하지 않았을 때보다는 비영요소수가 모두 감소하지만, israel과 seba에 대해서는 최소 차수 순서화를 적용했을 때보다 비영요소수가 증가하였다.

fit1p와 같이 밀집열이 완전밀집열인 경우일수록 밀집열분할에 의해서 추가되는 행의 차수가 최소 차수 순서화가 좋지 않은 결과를 만들어내도록 하고, israel이나 seba와 같이 밀집열이지만 영요소가 일부 있는 경우에는 최소 차수 순서화에 비교적 적은 영향을 미치는 것으로 보인다.

[표 4]는 순서화에 소요된 시간과 내부점 선형계획법을 수행하는 데에 걸린 전체 수행시간이다. 밀집열 분할을 수행하지 않았을 때에 순서화 수행 시간이 가장 짧게 소요되었다. 이는 행의 개수가 적을 뿐 아니라, 밀집열이 순서화 도중에 거대한 구별불능점을 형성하기 때문이다. 밀집열을 분할한 경우에는 최소 부족수 순서화가 순서화 수행 시간이 가장 많이 걸렸다. 수정된 최소 차수 순서화는 최소 차수 순서화보다 대체로 빠른 계산을 보이는데, 이는 밀집열 분할로 추가된 행들을 제외하고 순서화를 수행했기 때문이다.

최소 차수 순서화를 적용했을 때에 israel과 seba의 경우 가장 빠르지만, fit1p와 fit2p의 경우에는 수행도가 매우 떨어진다. 최소 부족수 순서화는 fit1p와 fit2p에서 매우 빠른 속도를 보여준다. 수정된 최소 차수 순서화는 fit1p에서는 최소 부족수 순서화를 적용하였을 경우보다 빠르지만, fit2p의 경우에는 계산 시간이 매우 많이 증가된 것을 볼 수 있다. fit1p에서도 순서화에 소요된 시간 차이를 고려하면, 매 회에 소요된 시간은 최소 부족수 순서화에 비해서 많이 소요되었음을 알 수 있고, 이와같은 현상이 fit2p에서는 매우 크게 나타났다.

4. 결 론

밀집열분할 방법은 밀집열을 가진 선형계획법 문제를 내부점 선형계획법을 적용하여 풀 때에 매우 효과적인 방법이다. 밀집열의 밀집도가 낮을 때에는 최소 차수 순서화가 효과적이며, 밀집열의 밀집도가 매우 높을 경우에는 최소 부족수 순서화가 효과적이다.

두 가지 방법의 장단점을 고려하여 적용한 수정된 최소 차수 순서화 방법은 밀집열의 밀집도가 매우 높은 문제에서 분행행렬 L의 추가요소를 많이 생성하여 최소 부족수 순서화 방법에 비해서 내부점 방법을 수행하는 데에 많이 시간이 소요되었다. 수정된 최소 차수 순서화 방법에서는 추가된 행을 무조건 제외시켰기 때문에 추가된 행에 의한 추가 비영요소가 고려되지 않았기 때문에, 밀집열을 분할한 회수가 많을 경우에 순서화 결과가 좋지 않을 수 있기 때문이다. 따라서,

[표 4] 수행 시간

문제 이름	순서화 수행 시간				전체 수행 시간			
	분할 않음	최소 차수 순서화	최소 부족수 순서화	수정된 최소 차수 순서화	분할 않음	최소 차수 순서화	최소 부족수 순서화	수정된 최소 차수 순서화
israel	0.01	0.06	2.27	0.09	1.78	1.36	3.70	1.58
seba	0.13	0.05	1.96	0.04	10.09	1.87	3.78	3.23
fit1p	0.37	2.91	38.52	0.06	69.78	120.36	51.91	27.12
fit2p	*	*	323.50	0.57	*	*	454.41	2,107.06

* : 메모리 소요량이 많아서 계산을 수행하지 못했음

수정된 최소 차수 순서화 방법에 대해서 기존의 행들에 적용할 순서화 방법과 밀집열 분할에 의해서 추가된 행들에 적용할 순서화 방법에 대한 추후 연구가 필요하다.

참 고 문 헌

[1] Adler, I., N. Karmarkar, M. G. C. Resende and G. Veiga, "An implementation of Karmarkar's algorithm for linear programming", *Mathematical Programming*, No.44(1989), pp.297-235

[2] Andersen, Erling D., Jacek Gondzio, Csaba Meszaros, Xiaojie Xu, "Implementation of interior point methods for large scale linear programming", Technical Report 1996. 3, Logilab, HEC Geneva, Section of Management Studies, University of Geneva

[3] Andersen, Knud D., "A modified Schur complement method for handling dense columns in interior point methods for linear programming", Technical Report, Odense University, Dec 1, 1994

[4] Choi, I. C., C. L. Monma and D. F. Shanno, "Further development of a primal-dual interior point method", *ORSA Journal on Computing*, No.2(1990), pp.304-311

[5] Gondzio, Jacek, "Splitting dense columns of constraint matrix in interior point methods for large scale linear programming", *Optimization*, Vol.24(1992), pp.285-297

[6] Gondzio, Jacek, Personal Communication

[7] Vanderbei, Robert J., "ALPO: Another Linear Program Optimizer", *ORSA Journal on Computing*, Vol.5, No.2(1993), pp.134-146

[8] Vanderbei, Robert J., "Splitting dense columns in sparse linear systems", *Linear Algebra Appl.*, No.152(1991), pp.107-117