QR분해를 이용한 선형계획법의 구현과 그 효율에 관하여

박찬규*, 성명기*, 박순달*

* 서울대학교 산업공학과

ABSTRACT

내부점 방법에서는 개선 방향을 대칭 양정치 행렬로 이루어지는 선형시스템의 해를 구해야 하고 단체법에서는 단체 승수나 진입열을 계산하는데 기저 행렬로 이루어지는 선형시스템을 풀게 된다. 본 연구는 내부점 기법과 단체법에서 나타나는 선형시스템을 QR분해를 통해 푸는 방법을 구현하고 이에 대한 결과를 제시하였다. QR분해 방법으로는 Givens가 제시한 방법을 사용했으며 단체법에서 rank-one 수정 방법을 사용하였다.

QR분해를 적용한 단체법의 경우는 상·하 분해를 단체법에 비해 많이 느리며 내부점 기법에 적용하면 촐레스키 분해보다 2~8배 정도 수행시간이 더 소요되었다. 그러나 내부점 기법에서 QR분해 방법이 촐레스키 분해 방법에 비해 보다 수치적으로 보다 정확하였다.

1. 서론

선형 시스템을 푸는 방법 중에 흔히 사용되는 방법에는 촐레스키 분해, LU분해, QR분해 방법 등이 있다.[1] 촐레스키 분해는 대칭 양정치 행렬로 이루어진 선형시스템의 해를 구하는데 사용되고 LU분해는 역행렬이 존재하는 선형시스템의 해를 구하는데 주로 사용된다. 위의 두 가지 상황 모두에서 사용될 수 있다.

내부점 방법에서는 매 단계마다 개선 방향을 구하기 위해 대칭 양정치 행렬로 이루어지는 선형시스템의 해를 구해야 한다. 대칭 양정치 시스템의 해를 구하는데는 주로 촐레스키 방법이 쓰이는데 이는 촐레스키 방법의 계산량이 QR분해의 계산량보다 작고 희소행렬인 경우 희소도(sparsity)를 이용하기가 쉽기 때문이다. 그러나 만약 주어진 선형 시스템 내에 서로 종속적인 행이 존재하는 경우 다시 말하면 주어진 행렬이 full rank가 아닌 경우에는 촐레스키 분해를 그대로 적용하는데는 문제가 있다. 또한 원래 주어진 행렬이 대칭 양정치성을 만족하더라도 내부점 해가 최적해에 접근하면 개선 방향을 구하기 위한 선형 시스템이 대칭 양정치성 면에서 매우 불안정해질 수 있다. 따라서 위와 같은 상황에 보다 수치적으로 안정적인 것으로 알려진 QR분해를 사용하는 것을 생각할 수 있는데 QR분해를 내부점 선형계획법에 구현하고 그 수행도를 분석하는 것이 본 연구의 첫 번째 목적이다.

단체법에서도 기저 행렬이 수치적으로 불안정한 경우에는 단체승수나 할인가를 정확하게 구하는 방법이 필요하다. 또한 단체법은 기저 행렬에 대한 LU분해를 구한 다음 특정 조건이 만족될 때까지 rank-one 수정을 통해 기저 행렬의 분해 요소 값을 적산형 형태로 저장하게 된다. 따라서 원래의 기저에서 rank-one 수정된 수가 많을수록 계산 오차는 커지게 된다. 이와 같은 상황에 보다 수치적으로 안정적인 QR분해 방법을 사용하는 방안을 고려할 수 있는데 QR분해를 단체법에 적용하여 수행도를 분석하는 것이 본 연구에서는 두 번째 목적이다.

2. QR분해

일반적으로 QR분해는 A행렬을 아래의 그림과 같은 두 개의 행렬의 곱으로 나타내는 분해 방법이다. 여기서 $Q = (q_{ij})$은 각 열이 서로 직교하며 놈(norm)이 1인 직교행렬이고 $R = (r_{ij})$은 상삼각행렬이다.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1n} \\ q_{21} & q_{22} & \cdots & q_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ q_{m1} & q_{m2} & \cdots & q_{mn} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & 0 & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & r_{nn} \end{bmatrix}$$

A를 QR로 분해하는 방법에는 Householder방법, Givens방법, 수정 Gram-Schmidt방법 등이 있다. 이 중에서 많이 사용하고 있는 Givens방법을 통해 QR분해를 살펴보자.

Givens방법은 벡터의 특정한 한 요소를 0으로 만드는 Givens회전행렬을 곱하여 벡터를 원하는 형태로 변환하는 방법이다. Householder방법이 벡터의 인접해 있는 여러 개의 요소를 동시에 0으로 만들 수 있는 반면에 Givens방법의 한번에 하나의 요소만을 0으로 만들 수 있다.

다음과 같은 벡터 x가 있을 때 벡터 x를 (i,j) 좌표 평면상에서 회전시킴으로써 j번째 요소를 0으로 만드는 방법을 생각해 보자. 즉 아래의 그림과 같이 x에 적절한 Givens 변환 행렬을 곱함으로써 x'을 만드는 방법을 고려해 보자.

$$x^T = [\, x_1, x_2, \cdots, x_i, \cdots, x_j, \cdots, x_n \,]$$
$$\Downarrow \quad J(i, j, \theta)$$
$$x' = [\, x_1, x_2, \cdots, x_i', \cdots, 0, \cdots, x_n \,]$$

위와 같은 변환을 수행하는 Givens 변환행렬 J(i,j,θ)은 다음과 같은 식을 통하여 구할 수 있다.

$$J(i, j, \theta) = \begin{array}{c} \begin{matrix} & & i & & j & \end{matrix} \\ \begin{bmatrix} 1 & & \vdots & & \vdots & \\ \cdots & & c & \cdots & s & \cdots \\ & & \vdots & & \vdots & \\ \cdots & & -s & \cdots & c & \cdots \\ & & \vdots & & \vdots & 1 \end{bmatrix} \end{array} \begin{matrix} \\ i \\ \\ j \\ \end{matrix}$$

단, $c = \dfrac{x_i}{\sqrt{x_i^2 + x_j^2}}$, $s = \dfrac{x_j}{\sqrt{x_i^2 + x_j^2}}$

여기서 Givens변환행렬은 i와 j번째 열만이 단위 행렬의 열과 다른 행렬이다. 따라서 Givens변환행렬이 벡터에 곱해질 때 영향을 받는 요소는 i번째 요소와 j번째 요소뿐인데 이 때 j번째 요소는 0이 된다.

Givens방법은 위에서 정의된 Givens변환행렬을 A에 차례차례 곱해 나감으로써 A행렬의 대각 원소 아래 부분을 0으로 만들어 A행렬에 대한 QR분해를 얻는다. 예를 들어 3×4 행렬을 Givens방법에 의해 QR분해하는 과정을 예로 들면 다음 그림과 같다.

$$\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow{J_{12}} \begin{bmatrix} x & x & x \\ 0 & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow{J_{13}} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ x & x & x \end{bmatrix} \xrightarrow{J_{23}} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ x & x & x \end{bmatrix} \xrightarrow{J_{14}}$$

$$\begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & x & x \end{bmatrix} \xrightarrow{J_{24}} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix} \xrightarrow{J_{34}} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & 0 \end{bmatrix}$$

대각 아래의 원소들을 모두 0으로 만든 후에는 다음과 같은 식이 성립하게 되어 Q, R를 구할 수 있게 된다.

$$J_{34} J_{24} J_{14} J_{23} J_{13} J_{12} A = R$$
$$Q = J_{12} J_{13} J_{23} J_{14} J_{24} J_{34}$$

다음으로 QR분해의 수정에 대해 알아보자.

A를 QR분해를 통해 A = QR를 얻은 A행렬이 수정되어 $\overline{A}$가 되었다고 하자. $\overline{A}$에 대해서 다시 QR분해를 통하여 $\overline{A} = \overline{Q}\,\overline{R}$를 얻을 수도 있으나 이는 많은 계산량을 요구하므로 A행렬에 대해 얻어 놓은 Q, R를 통해 좀더 효율적으로 $\overline{Q}$, $\overline{R}$를 계산해 내는 방법에 대해서 알아보자.

A행렬을 수정하여 $\overline{A}$를 얻고자 할 때 다른 여러 가지 경우가 있을 수 있으나 흔히 사용되는 다음과 같은 세 가지 경우를 생각해 볼 수 있다.

· A행렬에 계수[rank]가 1인 행렬을 더하는 경우
· A행렬에 새로운 행 또는 열이 첨가되는 경우
· A행렬에서 행 또는 열이 삭제되는 경우

여기서는 첫 번째 경우로서 A행렬에 계수가 1인 행렬을 더하여 수정된 행렬 $\overline{A}$를 얻을 때 $\overline{A}$의 QR분해를 효율적으로 구하는 방법에 대해서 알아보도록 하자. 즉, A = QR로부터 $\overline{A} = A + uv^T = \overline{Q}\,\overline{R}$, $u, v \in \mathbf{R}^n$을 만족하는 $\overline{Q}$, $\overline{R}$를 구하고자 하는 것이다.

먼저 다음의 식을 보도록 하자.

$$Q(A + u\ v^T) = R + w\,v^T$$
단, $w = Q^T u$

여기서 다음과 같은 Givens Rotation들을 생각해 보자.

$$J_1^T J_2^T \cdots J_{n-1}^T w = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$
단, $J_k = J(k,\ k+1,\ \theta_k)$

$$\alpha = \pm \| w \|_2$$

위에서 구한 $J_1^T J_2^T \cdots J_{n-1}^T$를 앞 식의 양변에 곱하면 다음과 같은 결과를 얻는다.

$$J_1^T J_2^T \cdots J_{n-1}^T (R + w\,v^T) = H + \alpha\, e_1 v^T = \overline{H}$$
단, $H = J_1^T J_2^T \cdots J_{n-1}^T R$

이 때 $\overline{H}$는 upper Hessenberg행렬로서 Givens회전행렬에 의해 쉽게 QR분해될 수 있다.

$\overline{H}$에 적당한 Givens회전행렬을 곱하여 $\overline{A}$의 QR분해의 $\overline{R}$를 구할 수 있다. 이를 식으로 표현하면 아래와 같다.

$$G_{n-1}^T \cdots G_2^T G_1^T \overline{H} = \overline{R}$$
단, $G_k = G(k,\ k+1,\ \phi_k)$

이상의 논의를 종합하면 수정된 행렬 $\overline{A}$의 QR분해는 다음과 같이 얻어질 수 있다.

$$\overline{A} = A + u\,v^T = \overline{Q}\,\overline{R}$$
단, $\overline{Q} = Q J_{n-1} \cdots J_2 J_1 G_1 G_2 \cdots G_{n-1}$

그러면 위의 방법을 사용하여 $\overline{A}$를 QR분해를 할 경우 소요되는 계산량을 알아보자. 먼저 w에 Givens 회전행렬을 곱하는 과정에서 n^2, $\overline{H}$를 구하는데 $6n^2$, 마지막으로 $\overline{R}$를 계산하는데 $6n^2$의 연산이 필요하여 모두 $13n^2$의 연산이 소요된다. 이는 $\overline{A}$를 바로 QR분해할 경우 요구되는 $O(n^3)$의 연산량에 비해서 훨씬 작음을 알 수 있다.

3. QR분해를 이용한 단체법

3.1 적용 방법

다음과 같은 문제가 있다고 하자.

$$\begin{aligned} Max \quad & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ s.t. \quad & A\,\mathbf{x} \le \mathbf{b} \\ & \mathbf{x} \ge \mathbf{0} \end{aligned}$$

여기서 A는 m×n행렬이고 $\mathbf{c}$와 $\mathbf{x}$는 n차 벡터, $\mathbf{b}$는 m차 벡터이다.

단체법이란 초기기저가능해를 찾은 후 이보다 목적함수 값을 증가시키는 방향을 구하여 인접한 기저로 계속 움직여 가서 결국 최적해를 구하게 되는 선형계획법 풀이 방법이다.

단체법 프로그램의 효율성은 기저역행렬 B의 보관 및 갱신 방법의 효율적인 구현에 있다. B의 보관 방법으로는 B^{-1}로 보관 방법과 이를 분해하여 보관하는 방법이 있다. B^{-1}로 보관하는 방법으로는 모든 요소를 보관하는 방법과 비영요소만을 보관하는 방법이 있다. B를 분해해서 보관하는 방법으로는 상·하 삼각행렬 분해 방법(LU decomposion), 촐레스키(Cholesky) 분해 방법, 그리고 QR분해 방법 등이 있다. 한편 각각의 방법에는 자료구조, 분해 및 갱신 방법에서 여러 가지 변형이 있다. 이 중에서 QR분해에 대해서 알아보기로 한다.

QR분해를 이용한 단체법은 기저행렬 B를 $B=QR$ 로 분해한 뒤 Q가 직교행렬이여서 $Q^{-1}=Q^{T}$라는 성질과 R이 상삼각행렬이라는 성질을 이용하여 단체법을 수행하는데 그 과정은 다음과 같다.

단계 1 초기기저가능해

초기기저를 구하고 이에 따른 Q와 R을 구한다.

단계 2 단체승수

$\boldsymbol{\pi}B=\mathbf{c}_{\mathrm{B}}$인데 $B=QR$이기 때문에 $\boldsymbol{\pi}QR=\mathbf{c}_{\mathrm{B}}$가 되므로 다음과 같이 단체승수 $\boldsymbol{\pi}$를 구한다.

$$\begin{aligned} \boldsymbol{\pi} &= \mathbf{c}_{\mathrm{B}} R^{-1} Q^{T} \\ &= \mathbf{w}\, Q^{T}, \quad \text{단 } \mathbf{w}R = \mathbf{c}_{\mathrm{B}} \end{aligned}$$

단계 3 진입변수 선택

모든 비기저변수에 대해 $\bar{c}_j=\boldsymbol{\pi}A_{\cdot j}-c_j$를 계산한다. 이 때 모든 $\bar{c}_j$가 $\bar{c}_j\ge 0$를 만족하면 현재 해가 최적이고 아니면 $c_s=Min\ \bar{c}_j$인 x_s가 진입변수이다.

단계 4 탈락변수 선정

먼저 $\bar{A}_{\cdot s}$를 다음과 같이 계산한다.

$$\begin{aligned} \bar{A}_{\cdot s} &= B^{-1}A_{\cdot s} \\ &= (QR)^{-1}A_{\cdot s} \\ &= R^{-1}Q^{T}A_{\cdot s} \end{aligned}$$

비율검정에 의해 탈락변수 x_r을 선정한다.

단계 5 Q, R 및 해 수정

$B=QR$에서 $\bar{B}=\bar{Q}\bar{R}$을 구하고자 하는데 $\bar{B}=B+(A_{\cdot s}-A_{\cdot r})\,\mathbf{e}_r^{\mathrm{T}}$이라는 사실을 이용하여 $\bar{Q}$와 $\bar{R}$을 $\bar{B}$에서 새로 구하지 않고 기존의 Q, R에서 구한다.[1)]

희소성을 이용하기 편한 Givens방법을 사용하는데 다음 식을 보자.

$Q^T(B+(A_{.s}-A_{.r})\mathbf{e}_r^T)=R+\boldsymbol{w}\,\mathbf{e}_r^T$ 단, $\mathbf{w}=Q^T(A_{.s}-A_{.r})$

$\boldsymbol{w}$에 Givens rotation 행렬을 곱하여 다음과 같이 바꾼다.

$$J_2^1 J_3^2 \cdots J_n^{n-1}\ \mathbf{w} = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

여기서 J_i^j는 $\boldsymbol{w}$의 j요소로 i요소를 0으로 만드는 Givens회전행렬을 의미한다.

위의 Givens회전행렬을 식(1)에 곱하면 다음과 같이 된다.

$J_2^1 J_3^2 \cdots J_n^{n-1}(R+\ \mathbf{w}\mathbf{e}_r^T)=H+\alpha\ \mathbf{e}_1\mathbf{e}_r^T=\overline{H}$ 단, $H=J_2^1 J_3^2 \cdots J_n^{n-1}R$

이 때 $\overline{H}$는 Givens회전행렬에 의해 쉽게 QR로 분해될 수 있다. 즉 행렬의 대각선 아래 한 요소만이 비영이므로 1열에서 (NROW-1)행까지 대각요소로 바로 아래 요소를 지우는 Givens회전행렬을 곱해서 $\overline{H}$가 상삼각행렬이 되도록 한다. 즉 $G_n^{n-1}\cdots G_3^2 G_2^1\overline{H}=\overline{R}$가 된다. 이 때 $\overline{Q}$는 다음과 같다.

$$\overline{Q}=QJ_n^{n-1}\cdots J_3^2 J_2^1 G_2^1 G_3^2\cdots G_n^{n-1}$$

해를 수정한 후 단계 2로 간다.

3.2 실험 결과

선형계획 프로그램의 실험 자료로 널리 사용되고 있는 NETLIB 문제들에 대한 실험 결과는 <표1>과 같다.

<표1> QR분해 프로그램과 LU분해 프로그램의 수행 시간 비교

실험기종 : SUN SPARC 10

문제명	문제 크기	QR 분해		상·하 분해	
		연산 회수	시 간	연산 회수	시 간
AFIRO	28 * 32	13	1.5	13	0.1
KB2	44 * 41	61	25.4	47	0.3
SC50B	51 * 48	29	13.1	29	0.2
SC50A	51 * 48	29	13.3	28	0.2
ADLITTLE	57 * 97	130	89.9	110	0.8
BLEND	75 * 83	104	333.9	92	1.1
SCSD1	78 * 760	287	718.0	156	4.2
RECIPE	92 * 180	27	67.4	27	0.6
SHARE2B	97 * 79	102	241.4	129	1.3
SC105	106 * 103	58	191.6	60	0.4
STOCFOR1	118 * 111	84	607.1	94	0.9
SHARE1B	118 * 225	270	2135.9	253	3.2
SCAGR7	130 * 140	81	696.1	128	1.2
BEACONFD	174 * 262	44	2120.7	36	2.6

QR분해 프로그램과 진입변수 선택 및 탈락변수 선정 방법이 같고 기저행렬을 보관하는 방법이 다른, 기저행렬을 상·하삼각행렬로 분해하여 보관하는 프로그램과 선회연산 횟수와 시간에 대해서 비교하였다.[2)]

<표1>에서 보다시피 QR분해를 이용한 프로그램이 상·하 분해를 이용한 프로그램보다 매우 느린데, 특히 문제가 크질 수록 더욱 느려지고 있다. 이러한 이유는 QR분해 방법에서 기저행렬 B의 한 열만이 바뀌었을 때 행렬 Q와 R의 수정시 많은 계산량을 요구하기 때문이다.

---

1) $\mathbf{e}_r$은 r번째 요소만 1이고 나머지는 0인 m차 벡터이다

2) 상·하분해를 이용한 선형계획 프로그램은 서울대학교 산업공학과 체계분석실에서 개발한 LPAKO v2.3이다.

4. QR분해를 이용한 내부점 선형계획법

4.1 적용 방법

다음과 같은 표준 형태의 선형계획법 문제가 있다고 하자.

$$\text{P}: \quad \begin{matrix} \text{Max} & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ s.t & \mathrm{A}\,\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{matrix} \qquad \text{D}: \quad \begin{matrix} \text{Min} & \mathbf{b}^{\mathrm{T}}\mathbf{y} \\ s.t & \mathrm{A}^{\mathrm{T}}\mathbf{y} - \mathbf{z} = \mathbf{c} \\ & \mathbf{z} \geq 0 \end{matrix}$$

위의 선형계획법 문제를 풀기 위한 내부점 방법에는 원방법, 쌍대 방법, 원쌍대 방법 등이 있고 각 방법에 대해 아핀(Affine) 변환을 사용하거나 또는 장벽법(Barrier Method)를 사용하게 된다. 이 중에서 자주 사용되는 방법으로는 쌍대 아핀법(Dual Affine Scaling Interior Point Method)과 원쌍대 장벽법(Primal-Dual Barrier Interior Point Method)이다.

쌍대 아핀법은 현재의 쌍대 가능 내부점 해에서 아핀 변환을 통해 쌍대 가능성을 유지하면서 쌍대 목적함수를 감소시키는 방향을 구하여 이 방향을 따라 현재의 내부점 해를 이동시켜 나간다. 매 단계마다 목적함수 값은 감소하게 되어 결국에는 쌍대 최적해로 수렴해 나가게 된다. 현재의 쌍대 가능해 y^k , z^k 가 주어져 있을 때 개선 방향을 구하는 식은 다음과 같다.

$$\mathbf{dy}^{\mathrm{k}} = -(\mathrm{AZ}^{-2}\mathrm{A}^{\mathrm{T}})^{-1}\,\mathbf{b}$$

$$\mathbf{dz}^{\mathrm{k}} = \mathrm{A}^{\mathrm{T}}\,\mathbf{dy}^{\mathrm{k}} \qquad \text{단,} \quad Z = \mathrm{diag}(z^k)$$

원쌍대 장벽법은 원가능, 쌍대가능인 내부점해에서 원가능, 쌍대가능을 각각 유지하면서 쌍대 간격을 감소시키도록 개선 방향을 구하여 이 개선 방향을 따라 현재의 해를 이동시켜 나가는 방법으로 결국 상보여유정리를 만족시키는 해를 찾아 가게 된다. 현재의 원가능인 내부점해 x^k , 쌍대가능인 내부점해 y^k , z^k가 있을 때 개선 방향을 구하는 식은 다음과 같다.

$$\mathbf{dy}^{\mathrm{k}} = -(\mathrm{AXZ}^{-1}\mathrm{A}^{\mathrm{T}})^{-1}(\,\mathbf{b} - \mu^{\mathrm{k}}\mathrm{AZ}^{-1}\,\mathbf{e})$$

$$\mathbf{dz}^{\mathrm{k}} = \mathrm{A}^{\mathrm{T}}\,\mathbf{dy}^{\mathrm{k}}$$

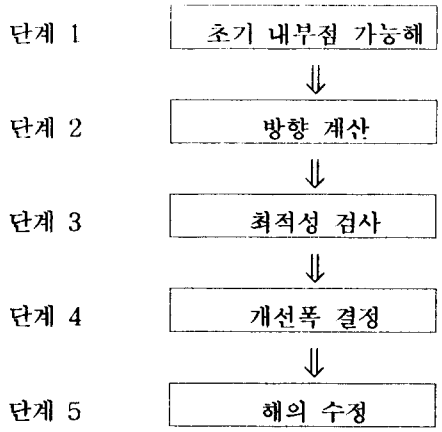
$$\mathbf{dx}^{\mathrm{k}} = \mu^{\mathrm{k}}\mathrm{Z}^{-1}\,\mathbf{e} - \mathrm{X}\,\mathbf{e} - \mathrm{XZ}^{-1}\,\mathbf{dz}^{\mathrm{k}}$$

$$\text{단,} \quad \mu^k = \rho\,\frac{\mathbf{e}^{\mathrm{T}}\mathrm{XZ}\,\mathbf{e}}{n}, \quad X = \mathrm{diag}(x^k), \quad Z = \mathrm{diag}(z^k)$$

위의 개선 방향을 구하는 식을 보면 두 방법 모두 다음과 같은 형태의 대칭 양정치 행렬로 이루어진 선형시스템의 해를 찾아야 됨을 알 수 있다. 이는 현재까지 알려진 내부점 방법에 공통적으로 적용된다.

$A\Theta A^T\,\mathbf{v} = \mathbf{u}$, Θ는 대각요소가 모두 양인 대각 행렬

다음은 QR분해 방법을 내부점 선형계획법에 실제로 어떻게 적용할 수 있는가에 대해 알아 보자. 내부점 선형계획법 해법에 의해 풀고자 할 때 내부점 선형계획법의 일반적인 흐름을 나타내면 다음 그림과 같다.



QR분해는 방향을 구하는 선형시스템을 푸는 한 가지 방법이므로 위의 다섯 단계 중에 촐레스키 분해 대신에 QR분해를 사용할 경우에 영향을 받는 부분은 방향을 결정하는 단계 3 부분이다. 그러면 내부점 선형계획법 해법의 하나인 쌍대 아핀법의 방향의 경우를 예로 들어 QR분해를 적용할 경우 방향을 구하는 방법을 보자. 현재의 쌍대 가능해 y^k , z^k가 주어져 있을 때 개선 방향을 계산하는 과정은 다음과 같다.

단계 3. 방향 계산

$$\mathbf{dy^k} = -(\mathrm{AZ^{-2}A^T})^{-1}\,\mathbf{b}$$

$$\mathbf{dz^k} = \mathrm{A^T}\,\mathbf{dy^k} \quad 단, \quad Z = \mathrm{diag}(z^k)$$

위의 두 번째 식은 단순한 행렬의 곱셈으로 구할 수 있으므로 중요한 문제가 되는 것은 첫 번째 식이다. 첫 번째 식에는 역행렬이 포함되어 있으므로 이를 효과적으로 해결하고자 하는 것이 QR분해와 Cholesky분해 방법을 쓰는 목적이다. 첫 번째 식의 $\mathbf{dy}^k$를 QR분해를 통해 구하는 절차와 Cholesky분해를 통해 구하는 방법을 정리하면 다음과 같다.

QR 분해의 경우	Cholesky 분해의 경우
① $Z^{-1}A^T$를 계산한다.	① $AZ^{-2}A^T$를 계산한다.
② $Z^{-1}A^T = QR$로 분해한다.	② $AZ^{-2}A^T = LL^T$로 분해한다.
③ $R^TR\ \mathbf{dy}^k = -\mathbf{b}$를 풀어 $\mathbf{dy}^k$를 구한다.	③ $R^TR\ \mathbf{dy}^k = -\mathbf{b}$를 풀어 $\mathbf{dy}^k$를 구한다.
④ $\mathbf{dz^k} = \mathrm{A^T}\,\mathbf{dy^k}$를 구한다.	④ $\mathbf{dz^k} = \mathrm{A^T}\,\mathbf{dy^k}$를 구한다.

QR분해를 수행하는 방법에는 앞절에서 언급한 것과 같이 Householder 방법, MGS 방법, Givens 방법 등이 있다. 또한 Givens 방법에도 행순서로 또는 열순서로 하느냐에 2가지로 나눌 수 있다. QR분해 중에서 Householder 방법이 수치적으로 안정적이고 계산량도 다른 QR분해 방법에 비해 작지만 희소도를 이용하기 어려운 특징이 있다. 물론 Q행렬을 보관해야 하는 경우에는 Householder 방법이 희소도면에서 유리하나 QR분해를 내부점 선형계획법에 적용할 때는 상삼각행렬인 R만 저장하면 되므로 가능하면 희소도를 충분히 이용하여 R를 구할 수 있는 방법을 사용해야 한다. 또한 기억 공간면에서도 QR분해를 구현할 때 실제로 필요한 기억 공간이 적어야 하고 구현하는데 필요한 자료구조도 간단한 방법을 사용해야 한다. 이러한 측면에서 볼 때 예비 실험 결과 Givens 방법을 행순서로 적용하는 것이 가장 효과적인 것으로 판명되었다. 따라서 이후의 논의에서는 내부점 선형계획법의 QR분해는 Givens 방법을 행순서로 적용하여 구하는 것으로 가정한다.

4.2 행의 순서화

주어진 행렬이 희소행렬인 경우 Givens 방법에 의해 R를 구할 때 실제 계산량은 주어진 행렬의 행 또는 열의 순서를 바꿈으로써 줄일 수 있다. 행의 순서를 바꾸는 것은 Permutation 행렬 P를 곱하는 것과 동일하므로 내부점 선형계획법의 개선 방향을 구하는 식은 다음과 같이 바뀐다. 여기서 $\overline{R}$는 $Z^{-1}A^{T}P^{T}$ 행렬을 QR분해했을 때의 결과로 나오는 상삼각 행렬이다.

$$\overline{R}^{T}\overline{R}P\ \mathbf{dy}^{k} = -P\mathbf{b}$$

QR분해의 결과로 나오는 상삼각 행렬 R과 촐레스키 분해 결과로 나오는 L^{T}는 동일한 행렬이므로 촐레스키 분해시 L의 비영 요소수를 줄이는 사용되는 순서화 방법들을 QR분해에서도 사용할 수 있다. 순서화 방법으로 가장 많이 사용되는 것이 최소 차수 순서화 방법인데 최소 차수 순서화 방법의 흐름은 다음과 같다. 최소 차수 순서화 방법에 대한 보다 자세한 설명은 []를 참조하기 바란다.

단계 1. 초기화

행렬 AA^{T}에 해당하는 무방향 그래프 G를 구한다.

단계 2. 최소 차수 노드 선택

G에서 최소 차수 노드 중에 하나를 임의로 선택하여 G에서 삭제한다.

단계 3. 삭제 그래프와 차수 수정

삭제 그래프를 수정하고 각 노드의 차수를 구한다.

단계 4. 종료 판정

G에 노드가 있으면 단계 2로, 아니면 마친다.

본 연구에서는 최소 차수 순서화 방법을 구현하기 위해 Liu가 제시한 Quotient 그래프 구조를 사용했으며 Indistinguishable 노드, 외부 차수(external degree) 등의 개념을 사용한다. 또한 최소 차수 노드간에는 임의로 한 노드를 택하여 삭제하는 방법을 사용하였다.

4.3 열순서화

Givens 방법을 행순서로 적용하는 경우 주어진 A행렬의 열의 순서를 바꾸는 것은 R행렬의 각 요소 값에는 전혀 영향을 미치지 않으나 R행렬의 계산하는데 실제 행해야 하는 연산량에는 큰 영향을 미치게 된다. 또한 중간 과정 생기는 비영요소의 수에도 영향을 미친다. 이에 대한 자세한 []를 참조하기 바란다. 주의해야 할 것은 A행렬의 열을 바꾸는 것은 Z-1AT의 행의 순서를 바꾸는 효과를 가져온다는 사실이다.

R을 구하는데 소요되는 연산량을 감소시킬 수 있는 열순서를 구하는 방법에는 논의는 [][]에서 다루고 있다. Duff는 주로 행과 열의 순서를 실제 Givens 방법을 통해 QR분해를 하는 과정에서 동적으로 결정하는 방법을 제안했다. George와 NG는 width-two separator를 통한 열의 순서화 방법을 제시하였으나 실제로 제시한 방법을 구현하기는 쉽지 않다. 위의 두 연구 모두 행의 순서가 미리 정해진 상태에서 열의 순서를 정하는 문제에 대해서는 다루고 있지 않다.

앞절에서의 설명한 최소 차수 순서화 방법에 의해 A행렬의 행의 순서가 정해져 있을 때 다음과 같은 열 순서화 방법들을 고려해 볼 수 있다.

① Maximum Row Subscript에 의한 방법

각 열에서 가장 나중에 나오는 비영요소의 행의 인덱스가 작은 순서대로 열순서화함을 의미한다. 이 방법은 Duff가 제시한 개념이다.

② Leading Row Subscript에 의한 방법

각 열에서 가장 처음에 나오는 비영요소의 행의 인덱스가 작은 순서대로 열순서화한다. 이 방법은 열

순서화 A행렬의 비영요소 구조가 George와 NG가 제시한 열순서화 방법으로 생기는 A행렬의 비영요소 구조와 비슷하도록 하기 위한 방법이다.

③ Column Count에 의한 방법

열의 비영 요소수(Column Count)가 작은 순서대로 열의 순서를 정하는 방법이다. 비영요소가 많은 열이 R에 추가되면 다음 단계에서 많은 계산을 초래할 것이기 때문이다.

4.4 실험 결과

① 열순서화 방법간의 비교

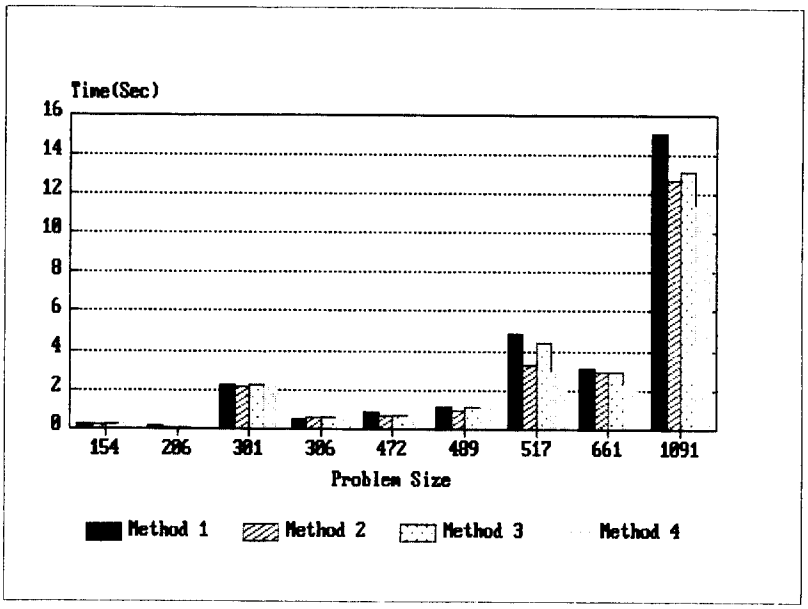
최소 차수 순서화 방법에 의해 미리 행순서를 정한 다음 앞절에서 제시한 3개의 열순서화 방법간의 계산 시간 비교는 다음 그림과 같다. 계산 시간은 Z-1AT를 QR분해하는데 소요되는 시간을 측정한 것이다. 문제 크기는 주어진 A행렬의 행의 갯수이다.

방법 1. 열순서화를 하지 않음.

방법 2. Maximum Row Subscript에 의한 방법

방법 3. Leading Row Subscript에 의한 방법

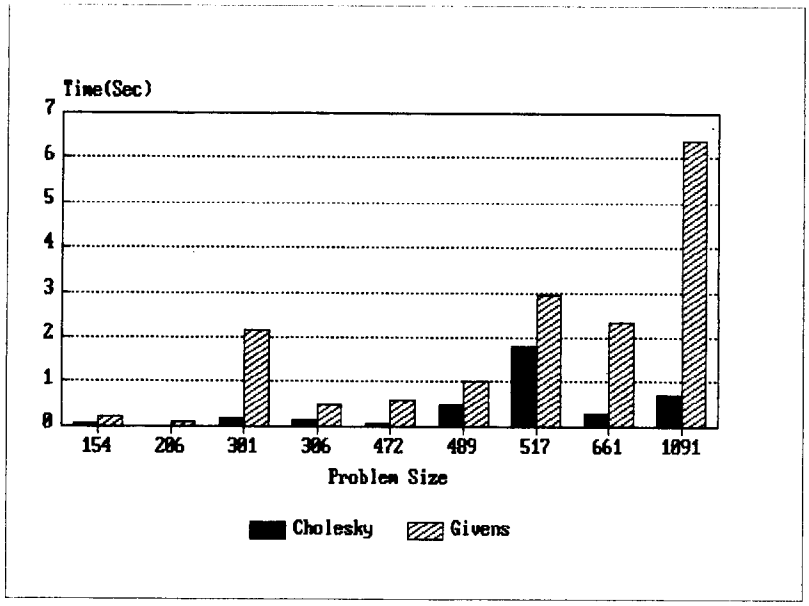
방법 4. Column Count에 의한 방법



실험 결과에 의하면 Column Count에 의한 방법이 다른 방법에 비해 우수함을 알 수 있다.

② 촐레스키 분해와 QR분해의 수행 시간 비교

개선 방향을 구하는 식에서 촐레스키 분해를 사용하여 L를 구하는 시간과 QR분해를 통하여 R를 구하는 시간을 비교하면 아래의 그림과 같다. 촐레스키 분해와 QR분해 모두 행순서화는 최소차수 순서화 방법을 사용했으며 QR분해 시의 열순서화에는 실험 결과 가장 우수한 것으로 나타난 Column Count에 의한 방법을 사용하였다. 문제 크기는 A행렬의 행의 갯수를 의미한다



그림에서 촐레스키 분해 방법이 QR분해 2~8배 정도 빠르다는 것을 알 수 있다.

③ 촐레스키 분해와 QR분해의 계산 오차 비교

계산 오차는 서로 종속인 행이 있는 문제를 촐레스키 분해 또는 QR분해하여 L(또는 R)에서 중복제약식이 있는 열(행)의 대각선 요소가 0에서 벗어난 정도를 측정하여 비교하였다. 서로 종속인 행이 있는 문제는 원문제의 행 2~4개를 조합하여 이들과 일차 종속인 새로운 행을 만들어 원문제에 첨가하면 쉽게 만들 수 있다. 4개, 5개 제약식간의 일차종속 관계가 성립할 때 분해 결과 해당 대각선 요소의 0에 벗어난 정도를 측정한 실험 결과는 다음 표와 같다. 또한 행·열순서화 방법은 각각 최소 차수 순서화 방법, Column Count에 의한 방법이다.

4개의 제약식이 일차 종속인 경우

문제크기	촐레스키	Givens
92×180	10e-07	10e-16
141×301	10e-07	10e-15
199×203	10e-08	10e-16
234×315	10e-07	10e-15
334×587	10e-08	10e-16
411×1000	10e-07	10e-15
498×614	0	10e-15
537×1775	0	10e-16
822×1571	10e-08	10e-15

5개의 제약식간에 일차 종속인 경우

문제크기	촐레스키	Givens
51×48	10e-08	10e-16
97×79	10e-08	10e-15
118×225	10e-07	10e-15
224×282	10e-07	10e-15
301×480	10e-08	10e-16
331×457	10e-08	10e-15
472×500	10e-07	10e-15
489×163	0	10e-14
525×854	10e-08	10e-17

Givens방법에 의한 QR분해가 촐레스키 분해보다 대체로 정확한 결과를 나타냄을 알 수 있다. 촐레스키 분해는 어떤 경우에는 Givens방법보다 보다 정확한 결과를 보이기도 하지만 최악의 경우 계산 오차가 10e-8 정도이지만 Givens방법은 거의 모든 문제에서 최악의 경우에도 10e-15 정도의 계산 오차만을 가지고 있어 Givens방법이 QR분해 방법에 비해 계산 오차 면에서 보다 안정적임을 알 수 있다.

5. 결론

QR분해를 적용한 단체법의 경우는 상·하 분해를 단체법에 비해 매우 느린데, 특히 문제가 크질 수록 더욱 더 느리다. 이러한 이유는 QR분해 방법에서 기저행렬 B의 한 열만이 바뀌었을 때 행렬 Q와 R의 수정시 많은 계산량을 요구하기 때문이다.

Givens 방법을 통한 QR분해를 내부점 기법에 적용하면 촐레스키 분해보다 2~8배 정도 수행시간이 더 소요된다. 이 때 QR분해를 위한 열순서화는 열의 비영요소를 이용한 방법이 우수한 것으로 나타났다. 또한 QR분해 방법이 촐레스키 분해 방법에 비해 보다 정확한 계산 결과를 가져온다. 따라서 중복행의 제거나 동적 축소등의 정확한 연산을 필요로하는 경우에는 QR분해 방법을 사용하는 것이 좋을 것으로 보인다.

6. 참고문헌

[1] 박순달, 선형계획법, 제3판, 1992, 민영사

[2] Alan George, Esmond NG, On Row and Column Orderings for Sparse Least Squares Problems, 1983, SIAM J. Numer. Anal. vol. 20, No. 2

[3] Alan George, Joseph W-H Liu, Computer Solution of Large Sparse Positive Definite Systems, 1981, Prentice-Hall, Inc., Englewood

[4] I.S. Duff, Pivot Selection and Row Ordering in Givens Reduction on Sparse Matrices, 1974, Computing 13

[5] G.H. Golub, C.F. Van Loan, Matrix Computations, 1983, John Hopkins University Press

[6] G.W. Stewart, On the Perturbation of LU, Cholesky and QR Factorization, 1993, SIAM J. Matrix Anal. Appl., Vol. 14, No. 4

[7] R.E. Marsten, M.J. Saltzman, D.F. Shanno, G.S. Pierce, J.F. Ballintijn, Implementation fo a Dual Affine Interior Point Algorithm for Linear Programming, 1989, ORSA Journal on Computing Vol 1, No. 4