# 大規模な非線形計画問題に対する<br>スパース準ニュートン更新について

指導教員　　福嶋雅夫　教授
山下信雄　助教授

Dinh Tien Hoang

京都大学大学院情報学研究科

数理工学専攻　修士課程

平成１７年４月入学


平成１９年２月

# 目 次

# 1 序論

非線形計画問題に対して現在，有効とされている手法にはSQP法と内点法がある [2, 13]．SQP法は各反復で元の問題を近似した二次計画問題を解くことによって次の点を生成する反復法である．一方，内点法は線形計画問題に対する解法として提案され，最近では一般の非線形計画問題に対しても拡張されている [7] ．内点法は各反復で元の問題のKKT条件を近似した線形方程式を解くことによって次の点を生成する．SQP法や内点法が各反復で解く (元の問題を近似した) 部分問題はラグランジュ関数のヘッセ行列またはその近似行列を用いて構成されている．そこで用いられる行列が正定値行列であれば，それらの手法は大域的収束することが知られている [13]．またその行列がヘッセ行列，またはそれによく近似されているときには，内点法とSQP法は超1次収束する．そのため，問題の性質に応じて，部分問題を構成する行列を適切に選ぶ必要がある．ヘッセ行列が容易に計算でき，そのヘッセ行列が正定値になる問題に対しては，ヘッセ行列を直接用いることが推奨されている．しかし，一般の非線形の問題ではヘッセ行列は正定値行列にならない．そのような場合では，信頼領域法を用いる手法が提案されているが，その手法の有効性はまだ完全に確立されていない．一方，ヘッセ行列が求まらないときやヘッセ行列が正定値とならないようなときには，正定値となる近似行列を用いる必要がある．このような近似行列を更新を準ニュートン更新という．非線形計画問題に対してよく利用される準ニュートン更新は修正BFGS更新である [13]．しかし，修正BFGS公式で更新された近似行列は密な行列になるため，大規模な問題には適用することができない．一般に大規模な最小化問題ではラグランジュ関数のヘッセ行列が疎になる．そこで，本論文では，その疎性を利用した新しい近似行列の生成方法を提案する．

最近，山下 [14] は制約なし最小化問題に対して，ヘッセ行列の疎性と正定値行列補完の結果を用いた新しい近似行列の更新手法 (Matrix Completion Quasi-Newton method, MCQN法) を提案している．山下は数値実験を行い，MCQN法が有効であることを報告している．本論文では山下のアイデアに基づいて，一般の非線形計画問題に対する準ニュートン更新を提案する．提案手法では，まず，修正BFGS法を用いてラグランジュ関数のヘッセ行列の疎構造に対応する部分の更新を行う．次に更新された部分の正定値補完行列を計算し，それを近似行列 $B_{k+1}$ の逆行列 $H_{k+1}$ とする．内点法やSQP法の部分問題を解くためには $H_{k+1}$ を含む線形方程式を解く必要がある．本論文ではその線形方程式の規模が大きいときには共役勾配法を用い，小さいときコレスキー 分解を用いることを提案する．このことによって疎構造を有効に利用することができ，各反復の計算量を大幅に削減することができる．

本論文は次のように構成される．第2節と第3節では，SQP法と内点法を簡単に紹介し，その中でよく利用されている修正BFGS公式とその欠点について述べる．第4節では，ラグランジュ関数のヘッセ行列の疎構造とそれに対応するグラフの性質に基づいた更新手法を提案する．さらに，提案した更新手法を用いた部分問題を解く手順や計算量について詳しく論じる．第5節では数値実験とその考察について述べる．第6節では本論文のまとめを行う．

本論文では次の記号を用いる．行列 $H$ に対して，$H_{:,\,j}$ は第 $j$ 列を表し，$H_{i,:}$ は第 $i$ 行を表す．また，集合 $S$ に対して，$|S|$ はその要素の数を表す．

# 2　SQP 法と内点法

非線形計画問題を解く解法には乗数法，逐次二次計画法 (SQP 法)，内点法などがある．乗数法と比べて，SQP 法と内点法は数値的に安定であり，現在多くの最適化パッケージに実装されている．本研究で考える準ニュートン更新は SQP 法と内点法の部分問題を構成するために用いられている．そこで，新しい準ニュートン更新の方法を提案する前に，この節では SQP 法と内点法について，簡潔に説明する．

## 2.1　SQP 法

以下の非線形計画問題を解くことを考える．

$$\begin{array}{ll} \min & f(x) \\ \text{s. t.} & c_i(x) = 0, i = 1, ..., m_e \\ & c_i(x) \leq 0, i = m_e + 1, ..., m \end{array} \tag{1}$$

ただし，目的関数 $f : R^n \mapsto R$ と関数 $c_i : R^n \mapsto R$, $i = 1, \ldots, m$ は 2 回連続的微分可能な関数とする．この問題のラグランジュ関数は

$$L(x, v) = f(x) + \sum_{i=1}^{m} v_i c_i(x)$$

となる．

SQP 法は毎回の反復で問題 (1) を近似した 2 次計画問題を解き，最終的に最適解に収束する点列を生成していく解法である．その概略は以下のとおりである．

**SQP 法**

**ステップ 0:** 適当な初期点 $x_0 \in R^n$ と正定値対称行列 $B_0 \in R^{n \times n}$ を選ぶ．$k = 0$ とする．

**ステップ 1:** 次の部分問題を解いて，その問題の解を探索方向 $d_k$ とする．

$$\begin{array}{ll} \min & \nabla f(x_k)^T d + \frac{1}{2} d^T B_k d \\ \text{s. t.} & c_i(x_k) + \nabla c_i(x_k)^T d = 0, i = 1, ..., m_e \\ & c_i(x_k) + \nabla c_i(x_k)^T d \leq 0, i = m_e + 1, ..., m \end{array} \tag{2}$$

**ステップ 2:** 適当な直線探索法を用いて，ステップサイズ $t_k$ を求める．$x_{k+1} = x_k + t_k d_k$ とする．

**ステップ 3:** 適当な方法で $B_{k+1}$ を更新する．$k = k + 1$ とし，ステップ 1 へ戻る．

SQP 法では行列 $B_k$ が正定値行列であり，下記で説明するステップサイズを用いれば, 大域的収束することが知られている．さらに $B_k$ がラグランジュ関数のヘッセ行列またはそれと近い行列であれば，SQP 法は速い収束性をもつ．特に，ヘッセ行列が常に正定値行列になる場合，$B_k$ としてヘッセ行列を用いると，二次収束すると知られている．しかし，一般にヘッセ行列は正定値行列にならないため，正定値行列となるヘッセ行列の近似行列を用いる必要がある．そのひとつの生成方法は修正 BFGS 更新式を用いる方法である．修正 BFGS 更新式については次節で説明する．

ステップ 2 では $t_k$ をもとめるために，次の $l_1$ ペナルティ関数を用いる．

$$F(x; r) = f(x) + r\left(\sum_{i=1}^{m_e} |c_i(x)| + \sum_{i=m_e+1}^{m} \max\{0, c_i(x)\}\right) \tag{3}$$

ここで，$r$ はペナルティパラメータである．$r$ が十分に大きいとき，$F(x; r)$ を最小化することは問題 (1) の KKT 点をもとめることと等価であると知られている．

$F(x;r)$ の方向 $d$ における方向微係数は以下に与えられる．

$$F_r(x;d) = f(x) + \nabla f(x)^T d + r\left(\sum_{i=1}^{m_e} |c_i(x) + \nabla c_i(x)^T d| + \sum_{i=m_e+1}^{m} \max\{0,\, c_i(x) + \nabla c_i(x_k)^T d\}\right)$$

このとき，以下の手順でステップサイズ $t_k$ を計算する．

**ステップ０：** $t_k := 1$ とおく．

**ステップ１：** 以下の式が成立するならば，終了する．

$$F(x_k + t_k d_k; r) \quad < \quad F(x_k; r) + \beta t_k (F_r(x_k; d_k) - F(x_k; r))$$

**ステップ２：** $t_k = \gamma t_k$ としてステップ１へ戻る．

ただし，$\gamma \in (0, 1)$ と $\beta \in (0, 1)$ は定数である．

二次計画問題 (2) を解く手法はいくつかあるが，ここでは有効制約法のひとつである Goldfard-Idnani の双対法 [4] を紹介する．

**Goldfard-Idnani の双対法**

**ステップ０：** $J = 1, ..., m_e$ とする．次の等式制約つき 2 次計画問題を解き，最適解 $d$ とそのラグランジュ乗数 $v$ を求める．

$$\begin{array}{ll} \min & \nabla f(x_k)^T d + \frac{1}{2} d^T B_k d \\ \text{s. t.} & c_i(x_k) + \nabla c_i(x_k)^T d = 0,\, i \in J \end{array}$$

**ステップ１：** $d$ が $J$ に含まれていないすべての不等式制約を満たすならば停止する．

**ステップ２：** 満たさない不等式制約をひとつ選んで，その制約の添え字記号を $s$ とする．主空間の探索方向 $z$ と双対空間の探索方向 $r$ を計算する．

$$\begin{array}{l} r = (A_J^T B_k^{-1} A_J)^{-1} A_J^T B_k^{-1} \nabla c_s(x_k) \\ z = B_k^{-1} \nabla c_s(x_k) - B_k^{-1} A_J r \end{array} \tag{4}$$

**ステップ３：** $r, z$ を用いて，$d$, $v$, $J$ を更新する．ステップ１へもどる．

このアルゴリズムでは最も計算を必要とするのはステップ２の $r$ と $z$ の計算である．

## 2.2 内点法

内点法は以下の問題を解くアルゴリズムである．

$$\begin{array}{ll} \min & f(x) \\ \text{s. t.} & c(x) = 0,\, x \geq 0 \end{array} \tag{5}$$

ただし，$c(x) = (c_1(x), ..., c_m(x))^T$ であり，$c_i : R^n \mapsto R$ である．なお，問題 (1) は $c_i(x) + z_i = 0$, $z_i \geq 0, i = m_e + 1, ..., m$ とおくと問題 (5) に帰着できる．従って，問題 (1) と問題 (5) は実質的に等価である．

問題 (5) のラグランジュ関数は以下のように与えられる．

$$L(\omega) = f(x) - y^T c(x) - z^T x$$

ここで，$y \in R^m$ と $z \in R^n$ はそれぞれ制約 $c(x) = 0$ と $x \geq 0$ に対応するラグランジュ乗数であり，$\omega = (x, y, z)^T$ である．

問題 (5) の KKT 条件は以下にようになる．

$$\begin{aligned} &\nabla_x L(\omega) \equiv \nabla f(x) - A(x)y - z = 0, \\ &c(x) = 0, \\ &x_i z_i = 0, \, (i = 1,\ldots, n), \\ &x_i \geq 0, \, z_i \geq 0, \, (i = 1,\ldots, n) \end{aligned} \tag{6}$$

ここで，

$$\begin{aligned} &X = \mathrm{diag}(x_1,\ldots, x_n) \in R^{n\times n}, \\ &Z = \mathrm{diag}(z_1,\ldots, z_n) \in R^{n\times n}, \\ &e = (1, ..., 1)^T \end{aligned}$$

とする．ただし，$\mathrm{diag}(x_1, ..., x_n)$ と $\mathrm{diag}(z_1, ..., z_n)$ はそれぞれ $x_1, ..., x_n$ と $z_1, ..., z_n$ を対角成分とする $n$ 次元対角行列である．すると，(6) は次のように書ける．

$$\begin{aligned} &r_0(\omega) \equiv \begin{pmatrix} \nabla_x L(\omega) \\ c(x) \\ XZe \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\ &x \geq 0, \, z \geq 0 \end{aligned}$$

問題 (5) に対して，次のバリア関数 $F_B(.;\mu) : R^n \mapsto R$ を考える．

$$F_B(x;\mu) = f(x) - \mu \textstyle\sum_{i=1}^{n} \log x_i, \quad (x \geq 0)$$

ただし，$\mu > 0$ はバリアパラメータである．そして，問題 (5) のかわりにバリア関数を用いた以下の問題を考える．

$$\begin{aligned} \min \quad & F_B(x;\mu) \\ \text{s. t.} \quad & c(x) = 0, x \geq 0 \end{aligned} \tag{7}$$

問題 (7) が解けるとき，その解を $x(\mu)$ とする．$\mu \to 0$ としたとき，$x(\mu)$ は問題 (5) の解に収束する．問題 (7) の KKT 条件は以下のように書ける．

$$\begin{aligned} &\nabla f(x) - A(x)y - \mu X^{-1} e = 0, \\ &c(x) = 0, \, x > 0 \end{aligned} \tag{8}$$

ただし，

$$A(x) = (\nabla c_1(x), ..., \nabla c_m(x)) = \begin{pmatrix} \frac{\partial c_1(x)}{\partial x_1} & \cdots & \frac{\partial c_m(x)}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial c_1(x)}{\partial x_n} & \cdots & \frac{\partial c_m(x)}{\partial x_n} \end{pmatrix} \in R^{n\times m}$$

である．ここで，

$$z = \mu X^{-1} e$$

とおくと，(8) は以下のように表わされる．

$$\begin{aligned} &r(\omega,\mu) \equiv \begin{pmatrix} \nabla f(x) - A(x)y - z \\ c(x) \\ XZe - \mu e \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\ &x > 0, \, z > 0 \end{aligned}$$

ここで，問題 (7) に対して，次のの $l_1$ ペナルティ関数を考える．

$$F(x;\mu) = f(x) - \mu \sum_{i=1}^{n} \log x_i + \rho \sum_{i=1}^{m} |c_i(x)|$$

ただし，$\rho > 0$ はペナルティパラメータである．

この関数 $F(\cdot;\mu)$ を最小化する以下の問題を考える．

$$\begin{array}{ll} \min & F(x;\mu) \\ \text{s. t.} & x \in R^n \end{array} \tag{9}$$

$\rho$ が十分に大きいときに，問題 (9) の停留点は問題 (7) の KKT 点になることが知られている [1, 15]．

主双対内点法は以下のようになる．

**内点法**

**ステップ 0:** $\varepsilon > 0$, $M_c > 0$,$\mu_0 > 0$ を選ぶ．$k = 0$ とおく．

**ステップ 1:** 次の不等式を満たす $\omega_{k+1}$ を求める．

$$\| r(\omega_{k+1},\mu_k) \| \leq M_c\mu_k \tag{10}$$

**ステップ 2:** もし $\| r_0(\omega_{k+1}) \| \leq \varepsilon$ ならば，終了する．

**ステップ 3:** $0 < \mu_{k+1} < \mu_k$ を満たす正数 $\mu_{k+1}$ を選ぶ．

**ステップ 4:** $k := k + 1$ とおいてステップ 1 にもどる．

問題 (10) を解くことができれば，アルゴリズムは大域的収束することが知られている [7]．内点法では部分問題 (10) を解くために，ニュートン法を用いる．つまり，次の $r(\cdot,\mu)$ 一次近似

$$r(\omega_k + \Delta\omega_k;\mu) \approx r(\omega_k;\mu) + \nabla r(\omega_k)\Delta\omega_k$$

を 0 にする $\Delta\omega_k$ を求める．ただし，

$$\nabla r(\omega) = \begin{pmatrix} \nabla_x^2 L(\omega) & -A(x) & -I \\ A^T(x) & 0 & 0 \\ Z & 0 & X \end{pmatrix}$$

である．

$\Delta\omega_k$ がペナルティ関数 $F(x;\mu)$ の降下方向になるためには，$\nabla_x^2 L(\omega)$ が正定値行列になる必要がある．しかし，一般に $\nabla_x^2 L(\omega)$ が正定値行列になるとは限らない．このようなときには，$\nabla_x^2 L(\omega)$ のかわりに正定値行列 $B_k$ を用いた次の方程式を解く．

$$J(\omega_k)\Delta\omega_k = -r(\omega_k;\mu) \tag{11}$$

ただし，

$$J(\omega_k) = \begin{pmatrix} B_k & -A(x_k) & -I \\ A(x_k)^T & 0 & 0 \\ Z_k & 0 & X_k \end{pmatrix}$$

である．

(11) の解 $\Delta\omega_k$ はペナルティ関数 $F(x;\mu)$ の降下方向になる．一方，内点法が速く収束するためには，$B_k$ はなるべく $\nabla_x^2 L(\omega)$ に近い行列になる必要がある．

部分問題 (11) を以下のように展開することによって，コンパクトな線形方程式に分解できる．

$$B_k\Delta x_k - A(x_k)\Delta y_k - \Delta z_k = -\nabla L(\omega_k),$$

$$A(x_k)^T \Delta x_k = -c(x_k),$$

$$Z_k \Delta x_k + X_k \Delta z_k = \mu e - X_k z_k$$

ここで，$B_k$ は正定値な行列であるため，$A(x_k)$ がフルランクならば，$\Delta x_k$,$\Delta y_k$,$\Delta z_k$ は以下のように計算できる．

$$\Delta y_k = (A(x_k)^T (X_k^{-1} Z_k + B_k)^{-1} A(x_k))^{-1} \left( -A(x_k)^T (X_k^{-1} Z_k + B_k)^{-1} (X_k^{-1} \mu e - z_k) - c(x_k) \right), \quad (12)$$

$$\Delta x_k = (X_k^{-1} Z_k + B_k)^{-1} A(x_k) \Delta y + (X_k^{-1} Z_k + B_k)^{-1} (X_k^{-1} \mu e - z_k) - c(x_k)), \quad (13)$$

$$\Delta z_k = -X_k^{-1} Z_k \Delta x_k + X_k^{-1} \mu e - z_k \quad (14)$$

この計算で最も時間を要するのは次の 2 つの計算である．

- $u = (X_k^{-1} Z_k + B_k)^{-1} v$
- $u = (A(x_k)^T (X_k^{-1} Z_k + B_k)^{-1} A(x_k))^{-1} v$

ただし，$v$ はあるベクトルである．

提案手法を用いて，これらの計算を行う方法は第 4 節で説明する．

# 3 修正 BFGS 公式とその欠点

前節で紹介したように SQP 法や内点法が大域的収束するためにはラグランジュ関数の近似ヘッセ行列 $B_k$ が正定値行列となる必要がある．さらに，速い収束を得るためには行列 $B_k$ がラグランジュ関数のヘッセ行列 $\nabla_x^2 L(\omega_k)$ に十分近似できていなければならない．

そのような $B_k$ の更新法としてよく用いられるものに，以下の BFGS 公式がある．

$$B_{k+1} = B_k + \frac{y_k (y_k)^T}{(y_k)^T s_k} - \frac{B_k s_k (s_k)^T B_k}{(s_k)^T B_k s_k} \quad (15)$$

ただし，

$$s_k = x_{k+1} - x_k \quad (16)$$

$$y_k = \nabla_x L(x_{k+1}, v_{k+1}) - \nabla_x L(x_k, v_k) \quad (17)$$

である．

しかし，ラグランジュ関数に対して一般には $y_k^T s_k > 0$ が成り立たないため，上式で更新される $B_k$ は必ずしも正定値行列になるとは限らない．そのような欠点を解決するため，Powell [10, 11] は式 (15) で $y_k$ に変わって，次の $\tilde{y}_k = \theta y_k + (1-\theta) s_k$ を使うことを提案した．ただし，

$$\theta := \begin{cases} 1 & s_k^T y_k \geq \alpha_k s_k^T B_k s_k \\ \frac{(1-\alpha_k) s_k^T B_k s_k}{s_k^T B_k s_k - s_k^T y_k} & \text{otherwise} \end{cases}$$

であり，$\alpha_k \in (0, 1]$ である．式 (15) で $y_k$ に変わって $\tilde{y}_k$ を用いる公式は**修正 BFGS 公式**と呼ばれる．

修正 BFGS 公式で生成された $B_k$ は密な行列になる．そのため，部分問題 (2) と (10) を解くときに，一般的に $O(n^3)$ の計算量が必要となる．また，$B_k$ を保存するのに，$O(n^2)$ のメモリが必要となる．そのため，修正 BFGS 法は小中規模の問題にしか適用できない．

そこで，本研究では $\nabla_x^2 L(\omega_k)$ とほぼ同様の疎性をもつような $B_k$ の更新方法を提案し，計算量やメモリスペースの削減を図る．

# 4　正定値行列補完を用いた準ニュートン更新とその実装方法

この節では，山下 [14] のアイデアに基づいて，$B_k$ が $\nabla_x^2 L(x,v)$ と同程度の疎性をもった行列となるような新しい更新方法を提案する．次に提案手法を用いた SQP 法と内点法の計算手順について述べる．その後に提案手法を用いることより，どのように計算量を削減できるかについて論じる．

## 4.1　正定値行列補完を用いた準ニュートン更新

ラグランジュ関数を

$$L(x,v) = f(x) + \sum_{i=1}^{m} v_i c_i(x)$$

とする．行列 $A$ に対して，集合 $\{(i,j)|A_{i,j} \neq 0\}$ を行列 $A$ の疎構造と呼ぶ．$B$ がある $x$ の行列関数であるときは $\{(i,j)|$ ある $x$ において $B(x)_{i,j} \neq 0\}$ を関数 $B$ の疎構造と呼ぶ．以下では $E$ をラグランジュ関数のヘッセ行列 $\nabla_x^2 L(x,v)$ の疎構造とする．

$$E = \left\{(i,\, j)|\ \text{ある } x \in R^n \text{と } v \in R^m \text{において } \nabla_x^2 L(x,v)_{i,\,j} \neq 0\right\}$$

次のように，正定値行列補完を定義する．

**定義 1.**　• 集合 $F \subseteq \{1,...,n\} \times \{1,...,n\}$ と $H_{i,j}, (i,j) \in F$ に対して，$K_{i,j} = H_{i,j}, \forall (i,j) \in F$ を満たす正定値行列 $K$ を $H_{i,j}, (i,j) \in F$ の**正定値行列補完**という．

従来の修正 BFGS 公式では $B_k$ を更新するが、本研究ではその逆行列 $H_k$ を更新することを考える．
さらに，以下の 2 つの条件を満たすような更新方法を提案する．

$$H_k \succ 0 (B_k \succ 0) \tag{18}$$

$$\text{ある集合 } F \supseteq E \text{ に対して, } (B_k)_{i,j} = 0, \forall (i,j) \notin F \tag{19}$$

条件 (18) はアルゴリズムの大域的収束性を保証するための条件である．条件 (19) では $B_k$ の疎構造がラグランジュ関数のヘッセ行列の疎構造 $E$ を含んでいることを意味している．つまり，$F$ は $E$ とほとんど同じ場合では，$B_k$ の疎構造がラグランジュ関数のヘッセ行列の疎構造とほとんど同じになる．条件 (18) と (19) を満たすように山下 [14] のアイデアに基づいた次の更新方法を提案する．

**提案の更新方法**

**前処理：** 集合 $F$ をもとめる．

**反復 $k$ での計算：** $H_k$ から以下の 2 段階で $H_{k+1}$ を計算する．

**第 1 段階：** 修正 BFGS 公式で $H_k$ から $(H_{k+1})_{i,j}, (i,j) \in F$ を計算する．

**第 2 段階：** $(H_{k+1})_{i,j}, (i,j) \in F$ の正定値行列補完を計算し，その行列を $H_{k+1}$ する．

### 4.1.1　前処理

前処理では，まず集合 $F$ をもとめる．さらに反復 $k$ で計算される $(H_{k+1})_{i,j}, (i,j) \in F$ の正定値行列補完を導出するための準備をする．これらの計算ではグラフ理論が必要となる．

集合 $E$ に対して，$V = 1,...,n$ を頂点集合，$E$ を辺としたグラフ $G(E,V)$ を考える．さらに，以下のグラフ理論の概念を定義する．

**定義 2.**
- 任意の 2 点が接続しているグラフを**完全グラフ**という．
- あるグラフの部分グラフで完全なものをそのグラフの**クリーク**という．
- 他のクリークの真の部分グラフにならないクリークを**極大である**という．さらに，あるグラフの極大クリークの集合をそのグラフの**極大クリーク族**という．
- その点と隣接している頂点がクリークを形成する点を**単体的**という．
- サイクル中でつながっていない 2 点を結ぶ辺を**コード**（弦）という．
- グラフ中の長さが 4 位上のすべてのサイクルがコードを持てば，このグラフを**コーダルグラフ**という．
- $G(V,F)$ がコーダルグラフで $F \supseteq E$ であるとき，$G(V,F)$ は $G(V,E)$ の**コーダル拡張**という．

以下のステップに従って**前処理の手順**を説明する．

**ステップ 1：** 疎構造 $E$ を求める．

**ステップ 2：** $G(V,E)$ のコーダル拡張 $G(V,F)$ を求める．

**ステップ 3：** $G(V,F)$ の極大クリーク族 $\{C_r|r=1,\ldots,l\}$ を求める．

ステップ 1 では疎構造をもとめるために，例えば自動微分法を用いる．

ステップ 2 では計算量を削減するため，なるべくサイズが小さいコーダル拡張をもとめたい．しかし，サイズが最小となるコーダル拡張を求める問題は NP 完全であることが知られている．これまでに，比較的サイズが小さいコーダル拡張を求めるいくつかのヒューリスティックが提案されている．例えば，Minimum Degree Ordering Algorithm(最小次数法) と Nested Dissection Ordering Algorithm である．最小次数法では $F$ のサイズが小さくなるようにグラフ $G(V,E)$ の頂点の順序を入れ換える．得られたグラフからコーダルグラフ $G(V,F)$ をもとめるために，例えば Parter[9] のアルゴリズムを用いれば良い．

ステップ 3 では極大クリーク族 $\{C_r|r=1,\ldots,l\}$ を求めるために，コーダルグラフの性質を用いる．その性質とは単体的な頂点をもつことである．さらに，コーダルグラフから単体的な頂点を消去した部分グラフもまたコーダルグラフになる．この単体的な頂点の消去によって $G(V,F)$ の頂点に順序付けをすることができる．この順序を**完全消去順序**と呼ぶ．コーダルグラフの完全消去順序をもとめるためには，例えば Maximum Cardinality Search アルゴリズム [12] を用いる．完全消去順序からアルゴリズム [8] を用いて，極大クリーク族 $\{C_r|r=1,\ldots,l\}$ を求めることができる．

さらに，次の Running intersection property と呼ばれる条件を満たすように極大クリーク $\{C_r|r=1,\ldots,l\}$ の順序を入れ換えることができる [8].

$$\exists s \geq r+1\colon\ C_r \cap (C_{r+1} \cup C_{r+2} \cup \ldots \cup C_l) \subsetneq C_s \tag{20}$$

Running intersection property を満たす $\{C_r|r=1,\ldots,l\}$ から添字集合 $\{S_r|r=1,\ldots,l\}$ と $\{U_r|r=1,\ldots,l\}$ を以下のようにもとめる．

$$S_r = C_r \setminus (C_{r+1} \cup C_{r+2} \cup \ldots \cup C_l),\ r = 1,\ldots,\ l$$
$$U_r = C_r \cap (C_{r+1} \cup C_{r+2} \cup \ldots \cup C_l),\ r = 1,\ldots,\ l$$

$\{S_r|r=1,\ldots,l\}$ と $\{U_r|r=1,\ldots,l\}$ は後の副節で説明する $(H_{k+1})_{i,j}, (i,j) \in F$ の正定値行列補完を導出するのに用いられる．

### 4.1.2 $(H_{k+1})_{i,j}, (i,j) \in F$ の計算

この副節では $H_k$ から $(H_{k+1})_{i,j}, (i,j) \in F$ を計算する方法について説明する．

修正 BFGS 公式 [10, 11] は行列 $B_k$ 全体の更新式である．しかし，提案手法では $B_k$ の逆行列 $H_k$ の $F$ に対応する部分のみを更新する．修正 BFGS 公式に基づいて，以下の式を用いることを提案する．

$$(H_{k+1})_{i,j} = (H_k)_{i,\,j} + \rho(\tilde{s}_k)_i(\tilde{s}_k)_j - \frac{(H_k y_k)_i(\tilde{s}_k)_j + (\tilde{s}_k)_i(H_k y_k)_j}{\tilde{s}_k^T y_k}, (i,j) \in F \tag{21}$$

ただし，

$$\begin{aligned}
&s_k = x_{k+1} - x_k, \\
&y_k = \nabla_x L(x_{k+1}, v_{k+1}) - \nabla_x L(x_k, v_k), \\
&\theta := \begin{cases} 1 & s_k^T y_k \geq \alpha_k y_k^T H_k y_k \\ \dfrac{(1-\alpha_k) y_k^T H_k y_k}{y_k^T H_k y_k - s_k^T y_k} & \text{otherwise} \end{cases} \\
&\rho = \frac{1}{\tilde{s}_k^T y_k} + \frac{y_k^T H_k y_k}{(\tilde{s}_k^T y_k)^2}, \\
&\tilde{s}_k = \theta s_k + (1-\theta) H_k y_k,
\end{aligned} \tag{22}$$

である．ただし，$\alpha_k \in (0, 1]$ である．

**補題 4.1.** $H_k \succ 0$ ならば，$(H_{k+1})_{C_r C_r} \succ 0, \forall r \in \{1, ..., l\}$ が成り立つ．

*Proof.* (22) から $\tilde{s}_k^T y_k > 0$ が成立することを容易に確認することができる．さらに $H_k \succ 0$ であるため，以下に与えられる行列 $K$ は正定値行列になる．

$$K = H_k + \rho \tilde{s}_k \tilde{s}_k^T - \frac{(H_k y_k)\tilde{s}_k^T + \tilde{s}_k (H_k y_k)^T}{\tilde{s}_k^T y_k}$$

従って，$K_{i,j} = (H_{k+1})_{i,j}, \forall (i,j) \in F$ である．さらに，$(C_r \times C_r) \in F, \forall r \in \{1, ..., l\}$ であり，$K$ は正定値対称行列であるため，$(H_{k+1})_{C_r C_r} \succ 0, \forall r \in \{1, ..., l\}$ が成り立つ． □

### 4.1.3 正定値行列補完の計算

この副節では，前処理でもとめた添字集合 $\{S_r | r = 1, \ldots, l\}$ と $\{U_r | r = 1, \ldots, l\}$ を用いて，$(H_{k+1})_{i,j}, (i,j) \in F$ の正定値行列補完を導出する．

まず，$S_1, S_2, ..., S_l$ の要素を順番にならべたもとを $\pi = \{v_1, v_2, ..., v_n\}$ とする．補題 4.1 より $H_k \succ 0$ ならば，$(H_{k+1})_{C_r C_r} \succ 0, \forall r \in \{1, ..., l\}$ が成り立つ．[3] より，行列式が最大となる $(H_{k+1})_{i,j}, (i,j) \in F$ の正定値行列補完 $H_{k+1}$ は条件 (18) と (19) を満たす．さらに，$H_{k+1}$ は以下の式で与えられる．

$$H_{k+1} = L_1^T L_2^T \ldots L_{l-1}^T D L_{l-1} \ldots L_2 L_1 \tag{23}$$

ここで，$L_r$ は以下に与えられる．

$$[L_r]_{i,j} = \begin{cases} 1 & i = j \\ (\overline{H}_{U_r U_r}^{-1} \overline{H}_{U_r S_r})_{i,j} & (i,\, j) \in U_r \times S_r \\ 0 & \text{それ以外} \end{cases}$$

$L_r$ は $\pi$ の順序で見ると下三角行列となる．

また，$D$ は $\pi$ の順序で見ると以下の各行列 $D_{S_rS_r}, r = 1, \ldots, l$ を対角にもつブロック対角行列である．

$$D_{S_rS_r} = \begin{cases} \overline{H}_{S_rS_r} - \overline{H}_{S_rU_r}\overline{H}^{-1}_{U_rU_r}\overline{H}_{U_rS_r} & r \leq l-1 \\ \overline{H}_{S_rS_r} & r = l \end{cases}$$

ただし，$(\overline{H})_{i,j} = (H_{k+1})_{i,j}, \forall (i,j) \in F$ である．

アルゴリズムの大域的収束性を保証するために $H_k$ が正定値行列である必要がある．以下の 2 つの定理では，$H_0 \succ 0$ を選ぶと提案手法で更新される $H_k$ が常に正定値行列になることを示す．

**補題 4.2.** $H_k \succ 0, (H_k^{-1})_{i,j} = 0, (i,j) \in F, k \geq 1$ ならば，$H_{k+1} \succ 0, (H_{k+1}^{-1})_{i,j} = 0, (i,j) \in F$ が成り立つ．

*Proof.* (22) から $\tilde{s}_k^T y_k > 0$ が成立することを容易に確認することができる．このとき，[14] の Theorem 2 より，$H_{k+1} \succ 0, (H_{k+1}^{-1})_{i,j} = 0, (i,j) \in F$ が成り立つ． □

**定理 4.1.** $H_0 \succ 0$ ならば，$H_k \succ 0, (H_k^{-1})_{i,j} = 0, (i,j) \in F, \forall k \geq 1$ が成り立つ．

*Proof.* 補題 4.2 より，$H_1 \succ 0, (H_1^{-1})_{i,j} = 0, (i,j) \in F$ となることを証明すればよい．$G(V,F)$ がコーダルグラフであるため，[3] の Lemma 2.7 と [3] の式 (2.16) より，$H_1$ は $(H_0)_{i,j}, (i,j) \in F$ の行列式が最大となる正定値行列補完になる．さらに，[6] の Theorem 2 より，$(H_1^{-1})_{i,j} = 0, (i,j) \in F$ である． □

## 4.2 提案手法を用いた SQP 法の計算手順

本節では提案更新手法を用いた SQP 法の具体的な計算手順について説明する．ここで，提案手法更新手法を用いた SQP 法を **MC-SQP 法** (Matrix Completion Sequential Quadratic Programming method) と呼ぶ．

**MC-SQP 法の計算手順**

**ステップ 0：** ラグランジュ関数のヘッセ行列の疎構造 $E$，$G(V,F)$ が $G(V,E)$ のコーダル拡張となる集合 $F$，クリーク族 $C_r, r \in \{1, ..., l\}$，集合 $S_r, U_r, r \in 1, ..., l$ を求める．初期点 $x_0$，小さな $\epsilon > 0$ を選ぶ．正定値対称な初期行列 $H_0 \succ 0$ を選び，$(H_0)_{i,j}, (i,j) \in F$ を保存する．$k = 1$ とする．

**ステップ 1：** 式 (23) で $H_k$ を計算する．

**ステップ 2：** 部分問題 (2) を解いて，検索方向 $d_k$ を計算する．$d_k < \epsilon$ ならば，終了．

**ステップ 3：** ステップサイズ $t_k$ を求める．$x_k = x_{k-1} + t_k d_k$ とする．

**ステップ 4：** 式 (21-22) で $(H_k)_{i,j}, (i,j) \in F$ を計算する．$k = k+1$ とおいて，ステップ 1 へ．

図 1 は MC-SQP 法で更新される $H_k$ の様子を表わしている．図では行列 $H_k$ の $F$ に対応する部分は $(H_{k-1})_F$ と同じ色で染めている．これは $H_k$ が $(H_{k-1})_F$ の正定値行列補完であることを表わしている．

ステップ 0，1 および 4 については前の副節で説明した．ステップ 3 では従来の手法と同様に $l_1$ ペナルティ関数 (3) に対して，直線探索を行うことによってステップサイズを計算する．以下ではステップ 2 の $H_k$ を用いた部分問題の解法について述べる．

提案手法では，$B_k$ ではなく $H_k$ を用いる．双対法で部分問題を解くときに $H_k$ に関連する計算は式 (4) に現れる次の 3 つである．

- $u = H_k v$
- $u = A_J^T H_k A_J v$

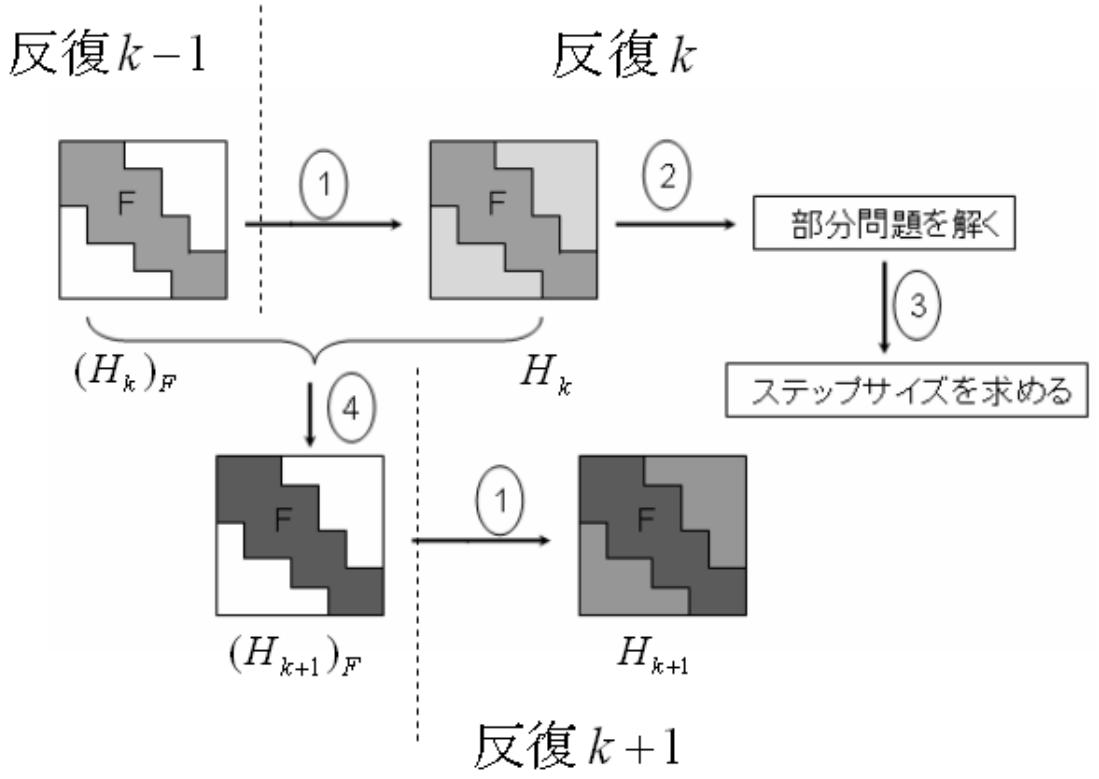


図 1: 提案手法を用いた SQP 法の計算手順

- $u = (A_J^T H_k A_J)^{-1} v$

ただし，$v$ はあるベクトルである．以下では簡単のために $H_k$ を $H$ と書く．

まず，$Hv$ について説明する．式 (32) より $Hv$ は以下のように書ける．

$$u = Hv = (L_1^T(L_2^T(\dots(L_{l-1}^T)(D(L_{l-1}(\dots(L_2(L_1 v))\dots)$$

ここで，

$$v_1 = L_1 v,\ v_2 = L_2 v_1,\ \cdots,\ v_{l-1} = L_{l-1} v_{l-2},\ v_l = D v_{l-1},\ v_{l+1} = L_{l-1}^T v_l,\ \cdots.,\ v_{2l} = L_1^T v_{2l-1}$$

の順番で計算する．

ここで，

$$[L_r v_j](i) = \begin{cases} v_j(i) & i \notin U_r \\ \sum_{k \in S_r} [L_{U_r S_r}](i,k) v_j(k) & i \in U_r \end{cases}$$

となる．また，$[D v_j](i) = \sum_{k \in S_r} [D_{S_r S_r}](i,k) v_j(k), \forall i \in S_r, r \in \{1,...,l\}$ と計算できる．

したがって，$Hv$ を計算するのに，$H$ を陽に計算する必要がなく，$L_{U_r S_r}, r = 1,...,l$ と $D_{S_r S_r}, r = 1,...,l$ を計算すれば良い．このとき，$L_r v_j$ の計算量は $O(|S_r||U_r|)$ となる．$D v_j$ の計算の計算量は $O(\sum_{r=1}^{l} |S_r|^2)$ となる．したがって，$Hv$ の計算量は $O(\sum_{r=1}^{l}(|U_r||S_r| + |S_r|^2))$ となる．大規模な問題では，一般的に $|U_r| = O(1), |S_r| = O(1)$ となることは多い．そのようなときは $l \leq n$ であるため，$Hv$ の計算量は $O(n)$ となる．

$u = A_J^T H A_J v$ の計算では式 (32) より以下のように書ける．

$$u = A_J^T H A_J v = (A_J^T(L_1^T(L_2^T(\dots(L_{l-1}^T)(D(L_{l-1}(\dots(L_2(L_1(A_J v))\dots)$$

本研究ではヤコビアン $A$ の疎構造を活用し，計算量削減をはかる．大規模な問題では $A_J$ の疎構造のサイズは $O(|J|)$ になることが多い．そのようなときでは，$A_J v$ の計算量は $O(|J|)$ となり，$u = A_J^T H_k A_J v$ の計算量は $O(n + |J|)$ となる．ただし，$|J| \leq m$ である．

$u = (A_J^T H A_J)^{-1} v$ の計算では以下のように $J$ のサイズによって，計算方法を選択することを提案する．

**$|J|^3 \geq n^2$ のとき** この場合，$|J|$ が大きいので，共役勾配法が適している．共役勾配法の各反復でもっとも計算量を必要とするのは $u = A_J^T H A_J v$ の計算である．

**$|J|^3 < n^2$ のとき** この場合，$|J|$ が比較的小さいので，$G = A_J^T H A_J$ を陽に計算し，$G$ をコレスキー分解する方法を用いる．

ここで，$|U_r| = O(1), |S_r| = O(1)$ のとき，$u = (A_J^T H A_J)^{-1} v$ の計算量について考える．

**$|J|^3 \geq n^2$ のとき** $u = A_J^T H A_J v$ の計算において，反復１回の計算量は $O(n + |J|)$ となる．また，正確な計算ができる場合，共役勾配法の反復回数は $|J|$ 以下になる．従って，$u = (A_J^T H A_J)^{-1} v$ の計算量は $O((n + |J|)|J|)$ 以下になる．

**$|J|^3 < n^2$ のとき** $G$ の計算において，$G$ の第 $j$ 列 $G_{:,j}$ は $G_{:,j} = A_J^T H((A_J)_{:,j})$ と計算できる．上で述べたようにこの計算量は $O(n+|J|)$ となる．従って，$G$ の計算量は $O((n+|J|)|J|)$ となる．また，$G$ のコレスキー分解の計算量は $|J|^3$ となる．従って，$u = (A_J^T H A_J)^{-1} v$ の計算量は $O((n + |J|)|J| + |J|^3)$ となる．

ここで，$|J|$ は有効制約の数であるため，$|J| \leq n$ となる．従って，$u = (A_J^T H A_J)^{-1} v$ が計算できる場合，その計算量は一般に $O(n^2)$ 以下になる．

### 4.3 提案手法を用いた内点法の計算手順

内点法の部分問題の解は式 (13)–(14) のように与えられる．部分問題を解くときに $B_k$ に関連する計算は式 (13) と式 (14) に現れる次の 2 つである．

- $u = (X_k^{-1} Z_k + B_k)^{-1} v$
- $u = (A(x_k)^T (X_k^{-1} Z_k + B_k)^{-1} A(x_k))^{-1} v$

ここで，簡単のために下付き文字 $k$ を省く．さらに，$A(x_k)$ を $A$ と書く．

まず，$u = (X^{-1} Z + B)^{-1} v$ の計算について考える．ここで，$B = H^{-1}$ であり，

$$B = L_1^{-1} L_2^{-1} \ldots L_{l-1}^{-1} D^{-1} (L_{l-1}^{-1})^T \ldots (L_2^{-1})^T (L_1^{-1})^T$$

である．ここで，$L_r^{-1}$ であり，

$$[L_r^{-1}]_{ij} = \begin{cases} 1 & i = j \\ -(\overline{H}_{U_r U_r}^{-1} \overline{H}_{U_r S_r})_{ij} & (i, j) \in U_r \times S_r \\ 0 & \text{それ以外} \end{cases}$$

である．また，$D^{-1}$ は $\pi$ の順序より見ると $(D_{S_r S_r})^{-1}, r = 1, \ldots, l$ を対角にもつブロック対角行列である．$|S_r| = O(1)$ の場合では，$(D_{S_r S_r})^{-1}$ は $O(1)$ の時間で計算できる．$X^{-1} Z + B$ は $B$ と同じ疎構造ともつ．さらに，$G(V, F)$ がコーダルグラフであることより，$X^{-1} Z + B$ は fill-in なしでコレスキー分解ができる．

$u = (A^T (X^{-1} Z + B)^{-1} A)^{-1} v$ の計算でもコレスキー分解を用いることを提案する．

### 4.4 提案手法の計算量について

この副節では提案手法を用いることによって，どのように計算量を削減することができるかについて論じる．

まず，記憶容量 (メモリ) について考える．本来の修正 BFGS 公式を用いる場合，$B_k$ を保存するために，$O(n^2)$ の記憶容量が必要となる．一方，提案手法を用いる場合は第 4 節で説明するように $B_k$ のかわりに，$(H_k)_{i,j}, (i,j) \in F$ と $H_k$ を更新する．しかし，実際では $H_k$ を記憶せず，$L_{U_rS_r}, D_{S_rS_r}, r = 1, ..., l$ を計算し，保存する．したがって，必要な記憶容量は $|F| + \sum_{r=1}^{l} |U_r||S_r| + |S_r|^2$ となる．$|F| = O(n), |U_r| = O(1), |S_r| = O(1)$ のときには，$l \leq n$ であるため，記憶容量は $O(n)$ になる．

次に，反復ごとの計算量について考える．修正 BFGS 公式で更新される $B_k$ を用いる場合では，有効制約法の 1 回の反復の計算量は $O(n^2) \sim O(n^3)$ となる．一方，提案方法を用いる場合では，前の副節で説明したとおり，各反復で一番計算量を必要とするのは $u = (A_J^T H_k A_J)^{-1} v$ の計算であり，$|F| = O(n)$, $|U_r| = O(1)$, $|S_r| = O(1)$ のときは $O(n^2)$ 以下になる．

## 5 数値実験

この節では新しい準ニュートン更新を用いた SQP 法の数値実験結果を報告する．実験環境は Linux，CPU 1674MHz，RAM 1GB のマシンである．メインプログラムは MATLAB6.5 で実装している．ただし，実行速度を向上するために，正定値行列補完を求める手順 1 ～ 4 のアルゴリズムや部分問題を解く有効制約法はすべて C 言語で実装し，MATLAB の MEX ファイルの形で実行する．

実験ではテスト問題集 CUTEr[5] の問題を解いた．CUTEr ではテスト問題集だけでなく，数値実験を支援するツールも添え付けられている．本研究では CUTEr のツールを用い，ヘッセ行列と制約のヤコビアンの疎構造を求めた．

各パラメータは次のように設定する．直線探索に用いる $\beta$ と $\gamma$ は $\beta = 0.1$, $\gamma = 0.8$ とした．また，準ニュートン更新に用いる $\theta_k$ は $\theta_k = 0.2$ とする．さらに，終了条件は以下の条件を満たすとき，プログラムを終了させた．

$$R = \max\{R_1, R_2, R_3, R_4, R_5\} \leq \sqrt{2} \times 10^{-6}, \tag{24}$$

ただし，

$$\begin{aligned} R_1 &= \| \nabla_x L(w) \|_1 / \max\{1, n \| \nabla f(x) \|\}, \\ R_2 &= \textstyle\sum_{j \in J_E} |c_j(x)| / |J_E|, \\ R_3 &= \textstyle\sum_{j \in J_I} |y_j c_j(x)| / |J_I|, \\ R_4 &= \textstyle\sum_{j \in J_I} |\min\{0, y_j\}|, \\ R_5 &= \textstyle\sum_{j \in J_I} |\min\{0, c_j(x)\}| \end{aligned}$$

である．また，2 時間以上経過しても条件 (24) を満たさない場合もプログラムを終了させた．

### 5.1 小中規模問題の数値実験

まず，変数と制約の数が 1500 以下の小中規模の問題に対して，提案した更新手法を用いた SQP 法と修正 BFGS 更新式を用いた SQP 法の実行時間と反復回数を比較した．実験で解いた問題は DTOC1L, DTOC1NA, DTOC1NB と DTOC1NC である．これらの問題はともに等式制約だけの問題で，目的関数は 4 次の多項式，制約関数は 2 次の多項式である．またこれらの問題は非常に疎な問題であり $|F| = n$，各制約関数のヤコビアンの非ゼロ要素の数は $10m$ 以下である．

実験結果を表 1 にまとめる．表 1 では loops は反復回数を表す．また，total time は前処理 (副節 4.1.1 で説明している) も含めた全体の実行時間 (秒単位) を表す．問題 DTOC1NA のところには # の記号がある

表 1: 小中規模な問題に対する実験結果

| Problem | (n,m) | loops | | total time(s) | |
|---|---|---|---|---|---|
| | | SQP | MC-SQP | SQP | MC-SQP |
| DTOC1L | (58，36) | 40 | 33 | 0.62 | 15.53 |
| | (298，196) | 38 | 45 | 77.48 | 183.10 |
| | (598，396) | 38 | 50 | 772.10 | 415.96 |
| | (745，490) | 34 | 37 | 1450 | 605.07 |
| DTOC1NA | (58，36) | 40 | 38 | 0.68 | 11.78 |
| | (298，196) | 40 | 42 | 95.86 | 109.56 |
| | (598，396) | 38 | 45 | 783.99 | 229.12 |
| | (745，490) | 34 | 40 | 1450 | 466.86 |
| | (1495，990) | # | 45 | # | 1080 |
| DTOC1NB | (58，36) | 41 | 38 | 0.69 | 11.27 |
| | (298，196) | 40 | 45 | 90.94 | 113.11 |
| | (598，396) | 41 | 50 | 848.54 | 243.97 |
| | (745，490) | 34 | 37 | 1580 | 664.72 |
| DTOC1NC | (58，36) | 23 | 23 | 0.37 | 5.58 |
| | (298，196) | 23 | 24 | 56.14 | 45.43 |
| | (598，396) | 23 | 24 | 484.14 | 95.31 |
| | (745，490) | 58 | 68 | 2510 | 911.36 |

が，これは既存手法を実行する最中にメモリ不足となり，実験が失敗することを表す．これらの表より，提案手法と既存手法はほとんど同じ反復回数で収束することが分かる．また，実行時間については，問題のサイズが大きくになるほど，提案手法の方が優れていることが分かる．

### 5.2 大規模問題の数値実験

次に変数の数と制約の数が数千程度の大規模な問題に対して，提案手法の有効性を確認した．提案手法が収束した問題の一覧は表 2 にまとめた．表 2 では $m_i = m - m_e$ が不等式制約の数を表す．loops は反復回数である．total time は全体の実行時間 (秒単位) を示し，loop time は前処理 (副節 4.1.1 で説明している) の実行時間を除いた SQP 反復だけの実行時間 (秒単位) を示す．これらの問題は全て疎な問題で不等式制約が少ない問題である．その中に問題 BLOCKQP1， BLOCKQP3， BLOCKQP5 は一つだけの不等式をもつ問題である．他の問題はすべて等式制約だけをもつ問題である．表 2 より，提案手法はかなり少ない反復回数で解が求まった．しかし，不等式が多い問題に対して，提案手法では実行時間が長くなり，効率が悪くなることが分かった．その原因は共役勾配法において適切な前処理をしなかったためだと思われる．

## 6 結論

本論文では大規模な非線形計画問題に対して，正定値行列補完を用いた準ニュートン更新手法を提案した．さらに SQP 法と内点法に提案手法を用いる際に，効率のよい計算手順を提案した．提案手法を用いた SQP 法を実装し，数値実験を行った．中小問題に対しては従来の修正 BFGS 法を用いたものと同程度の結果であった．また，修正 BFGS 法では扱うことができない大規模な問題を解くことができた．

表 2: 大規模問題に対する実験結果

| Problem | $n$ | $m_e$ | $m_i$ | loops | loop time(s) | total time(s) | $f(x)$ | $R$ |
|---|---|---|---|---|---|---|---|---|
| BLOCKQP1 | 6005 | 3000 | 1 | 2 | 7.03 | 44.29 | 2.4997 | 7.028e-13 |
| BLOCKQP3 | 6005 | 3000 | 1 | 2 | 14.75 | 53.10 | 2.4997 | 9.306e-13 |
| BLOCKQP5 | 6005 | 3000 | 1 | 2 | 14.71 | 52.81 | 2.4997 | 9.306e-13 |
| BRATU2D | 5184 | 4900 | 0 | 4 | 2865.15 | 2904.45 | 0 | 9.125e-13 |
| BRATU2DT | 5184 | 4900 | 0 | 4 | 2115.50 | 2153.12 | 0 | 9.709e-13 |
| BRATU3D | 4913 | 3375 | 0 | 4 | 322.82 | 352.13 | 0 | 7.816e-13 |
| BROYDN3D | 5000 | 5000 | 0 | 5 | 112.87 | 127.18 | 0 | 5.529e-11 |
| DTOC1L | 8998 | 5996 | 0 | 40 | 3996.52 | 4051.15 | 5.854 | 1.199e-06 |
| DTOC1NA | 8998 | 5996 | 0 | 40 | 2254.26 | 2392.31 | 6.148 | 1.200e-06 |
| DTOC1NB | 8998 | 5996 | 0 | 51 | 2822.43 | 2963.18 | 10.648 | 1.292e-06 |
| DTOC1NC | 8998 | 5996 | 0 | 10 | 529.76 | 669.21 | 52.777 | 0.701e-06 |
| DTOC1ND | 8998 | 5996 | 0 | 15 | 800.41 | 938.85 | 71.405 | 0.691e-06 |
| HUESTIS | 5000 | 2 | 0 | 2 | 3.47 | 7.77 | 1.73e+11 | 1.029e-06 |
| HUES-MOD | 5000 | 2 | 0 | 2 | 3.46 | 7.94 | 3.45e+07 | 1.013e-06 |
| MINC44 | 4227 | 4106 | 0 | 3 | 102.89 | 315.63 | 5.372e-05 | 9.854e-14 |
| MINPERM | 4227 | 4107 | 0 | 3 | 101.67 | 292.31 | 5.372e-05 | 8.008e-12 |
| SOSQP1 | 5000 | 2501 | 0 | 2 | 4.03 | 9.34 | -7.494e-11 | 5.750e-14 |
| YATP1SQ | 5040 | 5040 | 0 | 4 | 33.17 | 79.70 | 0 | 3.565e-07 |

本論文では提案手法を用いた SQP 法の数値実験を行ったが，提案手法を用いた内点法の数値実験は行っていない．内点法は，各反復で線形方程式を 1 回解くだけでよいので，SQP 法で解くことができなかった不等式制約を多くもつ問題に対して有効と考えられる．

**謝辞**

本論文を作成するにあたり，本論文全般に渡り，細部まで，数多くの御指摘，ご指導をいただいた山下信雄助教授に感謝いたします．ならびに，日頃から熱心に御指導をいただいた福嶋教授に感謝いたします．そして，日頃から数多くの助言をいただいた林助手に感謝いたします．また，大変お世話になった福島研究室の皆様に厚く御礼申し上げます．

# 参考文献

[1] R. Fletcher, Practical Methods of Optimization, Second edition, John Wiley & Sons, New York, 1987.

[2] 茨木俊秀, 福嶋雅夫, 最適化の手法, 共立出版株式会社, 1993.

[3] M. Fukuda, M. Kojima, K. Murota and K. Nakata, *Exploiting sparsity in semidefinite programming via matrix completion I: General framework*, SIAM Journal on Optimization, 11(2000), No. 3, pp. 647-674.

[4] D. Goldfarb and A. Idnani, *A numerically stable dual method for solving strictly convex quadratic programs*, Mathematical Programming, 27(1983), pp. 1-33.

[5] N. I. M. Gould, D. Orban and P. L. Toint, *CUTEr (and SifDec), a Constrained and Unconstrained Testing Environment, revisited*, Technical Report No. TR/PA/01/04.

[6] R. Grone, C. R. Johnson, E. M. Sa, and H. Wolkowicz, *Positive definite completions of partial hermitian planes*, Linear Algebra Appl., 58(1984), pp. 109-124.

[7] 小島政和, 土谷隆, 水野眞治, 矢部博, 内点法, 朝倉書店, 2001.

[8] 中村悟司, 行列補完を用いた半正定値問題の解法, 修士論文, 東京工業大学, 2001.

[9] S. V. Parter, *The use of linear graphs in Gause elimination*, SIAM Review, 3(1961), pp. 119-130.

[10] M. J. D. Powell, *A fast algorithm for nonlinearly constrained optimization calculation*, Numerical Analysis Proceedings, Dundee 1977, G. A. Watson, ed., Springer-Verlag, Berlin, 1978.

[11] M. J. D. Powell, *The performance of two subroutines for constrained optimization on some difficult test problems*, Numerical Optimization 1984, P. T. Boggs, R. Byrd and R. Schnabel, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1985.

[12] R. E. Tarjan and M. Yannakakis, *Simple linear-time algorithms to test chordality of graphs,test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs*, SIAM Journal on Computing, 13(1984), pp. 566-579.

[13] 矢部博, 八巻直一, 非線形計画法, 朝倉書店, 1999.

[14] N. Yamashita, *Sparse quasi-Newton updates with positive definite matrix completion*, Technical Report 2005-008, Applied Mathematics and Physics, Kyoto University (August 2005).

[15] H. Yamashita, *A globally convergent primal-dual interior point method for constrained optimization*, Optimization Methods and Software, 10(1998), pp. 443-469.