制約付大規模最適化問題に対する主双対内点法について

静岡大学工学部　鈴木 康司　SUZUKI Kouji
静岡大学工学部　八巻　直一　YAMAKI Naokazu
東京理科大学理学部　矢部　博　YABE Hiroshi

1　制約付き最適化問題

本研究で対象とする問題は，**非線形計画問題**（NonLinear Programming problem：以下 **NLP** と略す）である．本研究では NLP の中でも，特に大規模な制約付き NLP について取り扱う．

本節では，大規模な NLP を取り扱う前に，一般的な NLP について説明する．なお，NLP とは，問題中のいずれかの関数が非線形であるような問題である．本研究においては，次のような制約付き最適化問題を考える．

【制約条件付き最適化問題】

$$\begin{array}{lll} \min & f(\boldsymbol{x}), & \boldsymbol{x} \in \boldsymbol{R}^n \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}, & x_i \geq 0, i \in I_m \end{array} \tag{1}$$

ここで $f : \boldsymbol{R}^n \to \boldsymbol{R}$, $\boldsymbol{h} : \boldsymbol{R}^n \to \boldsymbol{R}^l$ は 2 回微分可能関数であり，I_m は $\{1, 2, \ldots, n\}$ の部分集合である．

また，$m = |I_m| > 0$, $D \in \boldsymbol{R}^{m \times n}$ をベクトル $\boldsymbol{e}_i^T, i \in I_m$ からなる行列とすると問題 (1) は次のように表すことができる．ただし，T は転置を表し，$\boldsymbol{e}_i \in \boldsymbol{R}^n$ は単位行列の i 列である．

【制約条件付き最適化問題】(本研究での定式化)

$$\begin{array}{lll} \min & f(\boldsymbol{x}), & \boldsymbol{x} \in \boldsymbol{R}^n \\ \text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}, & D\boldsymbol{x} \geq \boldsymbol{0} \end{array} \tag{2}$$

以下では簡略化のため

$$\boldsymbol{\xi} \equiv D\boldsymbol{x} \in \boldsymbol{R}^m$$

を用いる．

問題 (2) のラグランジュ関数は次のように定義される．

$$L(\boldsymbol{w}) = f(\boldsymbol{x}) - \boldsymbol{y}^T\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{z}^T\boldsymbol{\xi}$$

ただし $\boldsymbol{w} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ であり，$\boldsymbol{y} \in \boldsymbol{R}^l$，$\boldsymbol{z} \in \boldsymbol{R}^m$ はそれぞれ等式・非負制約に関するラグランジュ乗数ベクトルである．次に，Karush-Kuhn-Tucker(KKT) 条件は

$$\boldsymbol{r_0} = \begin{pmatrix} \nabla_{\boldsymbol{x}} L(\boldsymbol{w}) \\ \boldsymbol{h}(\boldsymbol{x}) \\ \Xi Z\boldsymbol{e} \end{pmatrix} = \boldsymbol{0}, \boldsymbol{\xi} \geq \boldsymbol{0}, \boldsymbol{z} \geq \boldsymbol{0} \tag{3}$$

である．ただし，

$$\begin{aligned}
\nabla_{\boldsymbol{x}} L(\boldsymbol{w}) &= \nabla f(\boldsymbol{x}) - \nabla \boldsymbol{h}(\boldsymbol{x})\boldsymbol{y} - D^T\boldsymbol{z}, \\
\nabla \boldsymbol{h}(\boldsymbol{x})^T &= \begin{pmatrix} \nabla h_1(x)^T \\ \nabla h_2(x)^T \\ \vdots \\ \nabla h_l(x)^T \end{pmatrix}, \\
X &= \mathrm{diag}(x_1, x_2, \ldots, x_n), \\
\Xi &= DXD^T = \mathrm{diag}(\xi_1, \xi_2, \ldots, \xi_m), \\
Z &= \mathrm{diag}(z_1, z_2, \ldots, z_m), \\
\boldsymbol{e} &= (1, \ldots, 1)^T \in \boldsymbol{R}^m.
\end{aligned}$$

また，$\mathrm{diag}(v_1, v_2, \cdots, v_n)$ は v_i を対角成分に並べた対角行列である．

ここで，内点法を用いて問題 (1) を解くために，次のログバリヤ関数の最小化問題を定義する．

【ログバリヤ関数の最小化問題】

$$\begin{array}{lll}
\min & f(\boldsymbol{x}) - \mu \displaystyle\sum_{i=1}^{m} \log(\xi_i), & \boldsymbol{\xi} \in \boldsymbol{R}_+^m \\
\text{s.t.} & \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}, & \\
\text{ただし} & \mu \text{ は正定数，} \boldsymbol{R}_+^m = \{\boldsymbol{\xi} \in \boldsymbol{R}^m | \boldsymbol{\xi} > \boldsymbol{0}\} &
\end{array} \tag{4}$$

十分小さな μ を用いた場合，問題 (4) の解は問題 (2) の近似解になる．問題 (4) のラグランジュ関数は，

$$L(\boldsymbol{w}) = f(\boldsymbol{x}) - \mu \sum_{i=1}^{m} \log(\xi_i) - \boldsymbol{y}^T \boldsymbol{h}(\boldsymbol{x})$$

であり，最適性条件は，

$$\boldsymbol{r}(\boldsymbol{w}, \mu) = \begin{pmatrix} \nabla_{\boldsymbol{x}} L(\boldsymbol{w}) \\ \boldsymbol{h}(\boldsymbol{x}) \\ \Xi Z\boldsymbol{e} - \mu \boldsymbol{e} \end{pmatrix} = \boldsymbol{0}, \boldsymbol{\xi} \geq \boldsymbol{0}, \boldsymbol{z} \geq \boldsymbol{0} \tag{5}$$

ただし，$\boldsymbol{y} \in \boldsymbol{R}^l$ は等式制約に関するラグランジュ乗数ベクトルであり，$\boldsymbol{z} \in \boldsymbol{R}^m$ は 3 番目の不等式を満たすために導入されたベクトルである．条件 (5) はバリヤ KKT 条件と呼ばれ，$\boldsymbol{w}(\mu) = (\boldsymbol{x}(\mu), \boldsymbol{y}(\mu), \boldsymbol{z}(\mu))$ はバリヤ KKT 条件を満たす．
$\rho \geq \|\boldsymbol{y}\|_\infty$ を満たす十分大きな ρ を用いれば，問題 (7) の最適性条件が条件 (5) になる．

1.1 主双対内点法

内点法 (内点ペナルティ法) の原理は，制約条件が付いた問題を考えるとき，実行可能領域内での関数値が，領域の境界に近づくにつれて大きくなり，境界上では無限大になるように目的関数 $f(\boldsymbol{x})$ を拡張して，その拡張関数の最小点を無制約最小化法を用いて求めることにある．内点法は，LP(Linear Programming) 問題や QP(Quadratic Programming) 問題に対して理論的にも実用的にも非常に活発に研究されてきた．LP 問題に対する内点法の中でも，近年，特に主双対内点法 (Primal-dual Interior Point method) が有望視されている．多くの研究者によって主双対内点法の大域的収束性や局所的収束性が研究されており，その実用性も高く評価されている．

そうした状況の中で，非線形最適化問題に対する内点法の研究が現在進行中である．これは，古典的な内点ペナルティ法の欠点を解消し，かつ，LP 問題で成功した内点法のアイデアを盛り込んだものである．

ここでは，制約付き最適化問題 (2) を扱う．主双対内点法では，最適性条件 (3) に非負パラメータ μ(これをバリヤパラメータという) を導入して，相補条件 $\Xi Z\boldsymbol{e} = \boldsymbol{0}$ を $\Xi Z\boldsymbol{e} = \mu\boldsymbol{e}$ で置き換えた方程式 (5) を考える．この条件はバリヤ KKT 条件と呼ばれ，これを満たす点 $\boldsymbol{w}(\mu) = (\boldsymbol{x}(\mu), \boldsymbol{y}(\mu), \boldsymbol{z}(\mu))$ を μ ーセンターという．また，μ を正の値から 0 に近づけたときに生成される μ ーセンターの軌跡をセンターパスという．$\mu = 0$ のとき，バリヤ KKT 条件 (5) が，もとの最適化問題の KKT 条件 (3) になる．

主双対内点法の基本的な考え方は，正数 μ が与えられたときに，μ ーセンターの近似解を求め，$\mu \rightarrow 0$ として最終的に KKT 点を求めていくのである．その際，速い収束性を実現するためにニュートン法を用いて非線形方程式 (5) を解き，変数 $(\boldsymbol{\xi}_k, \boldsymbol{z}_k)$ が非負条件に関して内点になるようにステップ幅を調整する．すなわち，$\Delta\boldsymbol{w}_k = (\Delta\boldsymbol{x}_k, \Delta\boldsymbol{y}_k, \Delta\boldsymbol{z}_k)^T$

をニュートン方向としたとき，k 回目の反復におけるニュートン方程式は

$$\begin{pmatrix} \nabla^2_{\boldsymbol{x}} L(\boldsymbol{w}_k) & -\nabla \boldsymbol{h}(\boldsymbol{x}_k) & -D^T \\ \nabla \boldsymbol{h}(\boldsymbol{x}_k)^T & \boldsymbol{0} & \boldsymbol{0} \\ Z_k D & \boldsymbol{0} & D X_k D^T \end{pmatrix} \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta \boldsymbol{y}_k \\ \Delta \boldsymbol{z}_k \end{pmatrix} = - \begin{pmatrix} \nabla_{\boldsymbol{x}} L(\boldsymbol{w}_k) \\ \boldsymbol{h}(\boldsymbol{x}_k) \\ D X_k D^T Z_k \boldsymbol{e} - \mu \boldsymbol{e} \end{pmatrix}$$

となる．このとき，$\Delta \boldsymbol{\xi}_k = D \Delta \boldsymbol{x}_k$ としたとき

$$\begin{aligned} \boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k + \alpha_k \Delta \boldsymbol{\xi}_k > \boldsymbol{0} \\ \boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \alpha_k \Delta \boldsymbol{z}_k > \boldsymbol{0} \end{aligned}$$

となるようにステップ幅 α_k が選ばれる．主双対内点法と従来のニュートン法との本質的な違いは，扱う方程式 (5) に摂動項 $\mu \boldsymbol{e}$ が含まれていること，点列 $(\boldsymbol{\xi}_k, \boldsymbol{z}_k)$ が非負条件に関して内点になるように減速をしなければならないことである．準ニュートン法のようにラグランジュ関数のヘッセ行列 $\nabla^2_{\boldsymbol{x}} L(\boldsymbol{w}_k)$ を行列 B_k で近似することも考えれば，ニュートン方程式に対応する連立１次方程式

$$J_k \Delta \boldsymbol{w}_k = -\boldsymbol{r}(\boldsymbol{w}_k, \mu) \tag{6}$$

を解いて探索方向 $\Delta \boldsymbol{w}_k = (\Delta \boldsymbol{x}_k, \Delta \boldsymbol{y}_k, \Delta \boldsymbol{z}_k)^T$ を求める解法が得られる．ただし，

$$J_k = \begin{pmatrix} B_k & -\nabla \boldsymbol{h}(\boldsymbol{x}_k) & -D^T \\ \nabla \boldsymbol{h}(\boldsymbol{x}_k)^T & \boldsymbol{0} & \boldsymbol{0} \\ Z_k D & \boldsymbol{0} & D X_k D^T \end{pmatrix}$$

である．ここで，$B_k = \nabla^2_{\boldsymbol{x}} L(\boldsymbol{w}_k)$ の場合がニュートン法に基づく主双対内点法，B_k をヘッセ行列 $\nabla^2_{\boldsymbol{x}} L(\boldsymbol{w}_k)$ の近似行列にとった場合が準ニュートン法に基づく主双対内点法になる．

なお，探索方向に対するステップ幅の決め方は，$(\boldsymbol{\xi}_k, \boldsymbol{z}_k)$ が内点になることばかりではなく，大域収束性を保証するために，直線探索法を用いて適当なメリット関数値を減少させることも要求される．Yamashita[8] は，ログバリヤ関数と l_1 型ペナルティ関数とを組み合わせて，メリット関数 (7) を提案した．

$$F(\boldsymbol{x}, \mu, \rho) = f(\boldsymbol{x}) - \mu \sum_{i=1}^{m} \log(\xi_i) + \rho \sum_{i=1}^{l} |h_i(\boldsymbol{x})| \tag{7}$$

この関数をバリヤペナルティ関数 (barrier penalty function) と呼ぶ．ここで第２項は非負条件に対するログバリヤ項で，$\boldsymbol{\xi} = \boldsymbol{0}$ の境界に近づかないように調整するもので，内点法にとっては本質的な項である．第３項は，$\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}$ に対する l_1 型ペナルティ項である．関数 (7) をメリット関数に用いる正当性は，十分に大きな ρ に対して，バリヤペナルティ関数を最小化する問題の最適性の必要条件がバリヤ KKT 条件であることによる．また，直線探索では Wolfe の基準が使われる．すなわち $\Delta \boldsymbol{\xi} = D \Delta \boldsymbol{x}$ としたとき $F(\boldsymbol{x} + \Delta \boldsymbol{x}, \mu, \rho)$ の１次近似

$$\begin{aligned} \hat{F}(\boldsymbol{x}, \Delta \boldsymbol{x}, \mu, \rho) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T \Delta \boldsymbol{x} - \mu \sum_{i=1}^{m} \left(\log(\xi_i) + \frac{\Delta \xi_i}{\xi_i} \right) \\ + \rho \sum_{j=1}^{l} |h_j(\boldsymbol{x}) + \nabla h_j(\boldsymbol{x})^T \Delta \boldsymbol{x}| \end{aligned}$$

に対して，

$$\Delta F(\boldsymbol{x}, \Delta\boldsymbol{x}, \mu, \rho) = \hat{F}(\boldsymbol{x}, \Delta\boldsymbol{x}, \mu, \rho) - F(\boldsymbol{x}, \mu, \rho)$$

と定義したとき，Armijo の基準は

$$F(\boldsymbol{x}_k + \alpha_k \Delta\boldsymbol{x}_k, \mu, \rho) \quad \leq \quad F(\boldsymbol{x}_k, \mu, \rho) + \zeta\alpha_k \Delta F(\boldsymbol{x}, \Delta\boldsymbol{x}, \mu, \rho) \tag{8}$$

となる．ただし，α_k は $\Delta\boldsymbol{x}_k$ 方向のステップ幅，ζ は $0 < \zeta < 1$ となる定数である．

アルゴリズムの概要は，内部反復では固定した正の数 μ に対して，上述の手順を $\| \boldsymbol{r}(\boldsymbol{w}_k, \mu) \|$ が十分に小さくなるまで繰り返し，外部反復では μ の値を徐々に 0 に近づけていくものである．以上をまとめると，次のアルゴリズムを得る．

【主双対内点法 (プロトタイプ)】

Step 0. $\mu > 0$ と $0 < \tau < 1$ を与える．

Step 1. 以下の内部反復によって，$\| \boldsymbol{r}(\boldsymbol{w}, \mu) \|$ が十分に小さくなるような w を見つける：

- Step 1.0 初期点 $\boldsymbol{w}_0$ と初期行列 B_0 を与える．$k = 0$ とおく (通常は前回の外部反復で得られた点 $\boldsymbol{w}$ を $\boldsymbol{w}_0$ に選ぶ).
- Step 1.1 連立 1 次方程式 (6) を解いて探索方向 $\Delta\boldsymbol{w}_k$ を求める．
- Step 1.2 $\boldsymbol{\xi}_k + \alpha_k \Delta\boldsymbol{\xi}_k > \boldsymbol{0}, \quad \boldsymbol{z}_k + \alpha_k \Delta\boldsymbol{z}_k > \boldsymbol{0}$ となり，かつ Armijo の基準 (8) を満足するステップ幅 α_k を求める．
- Step 1.3 $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \Delta\boldsymbol{x}_k, \quad \boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \alpha_k \Delta\boldsymbol{y}_k, \quad \boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \alpha_k \Delta\boldsymbol{z}_k$ とおく．
- Step 1.4 行列 B_{k+1} を求める．
- Step 1.5 $k := k + 1$ とおいて内部反復の Step1.1 へいく．

Step 2. もし μ の値が十分に小さければ，求まった $\boldsymbol{w} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ を解とみなして終了する．さもなければ，$\mu := \tau\mu$ とおいて外部反復の Step 1 へいく．

1.2 Maratos 効果

主双対内点法はニュートン方の考え方に基づいた数値解法であるので，局所的に速い収束を達成するためには，最終的に解の近くで，ステップ幅 $\alpha_k = 1$ が選ばれることが望ましい [1]．他方，大域収束性を保証するためにはメリット関数の減少を課さなければならず，そのために $\alpha_k = 1$ が採用されないことが起こりうる．この現象は Maratos 効果（Maratos effect）[2] と呼ばれる．Maratos 効果は，メリット関数である l_1 型正確なペナルティ関数が微分可能でないことと 2 次計画部分問題で非線形制約条件を線形化したことによって生じる．そこで，大域的収束を保持しながら，いかにして $\alpha_k = 1$ を採用して超 1 次収束性

を実現するか（Maratos 効果を避けるか）が，いろいろと研究されてきた．代表的な対応策として，非単調直線探索法を利用する方法，微分可能な正確なペナルティ関数をメリット関数として用いる方法などがあげられる．しかしながら，理論的な裏付けはともかくとして，実際の問題で Maratos 効果が起こることはほとんどない．

1.3　非単調ステップ

主双対内点法では，バリヤペナルティ関数に l_1 型正確なペナルティ関数項が含まれているので Maratos 効果の問題が起こり得る．この問題点を克服するため，本研究では主双対内点法に非単調ステップを組み込んでいる．非単調ステップとは目的関数値を単純に減少させるのではなくて，関数の増加をある程度許して数回の反復ごとに目的関数値を減少させてステップ幅 1 が採用されるチャンスを増やす手法である．メリット関数にバリヤペナルティ関数 (7) を用いた非単調主双対内点法のアルゴリズムを以下に示す．

【アルゴリズム 非単調主双対内点法】

Step 0. 初期パラメータ $\rho > 0$，$M_c > 0$ を選び，初期点 $\boldsymbol{w}_0 \in \boldsymbol{R}^n_+ \times \boldsymbol{R}^l_+ \times \boldsymbol{R}^m_+$ と $\|\boldsymbol{r}(\boldsymbol{w}_0, \mu_{-1})\| \le M_c\mu_{-1}$ を満たす $\mu_{-1} > 0$ を与える．$\lambda_0 = F(\boldsymbol{x}_0, \mu_{-1})$，$k = 0$ とおく．

Step 1. もし $\|\boldsymbol{r}_0(\boldsymbol{w}_k)\| \le \epsilon$ ならば停止する．さもなければ $\mu_k \in (0, \mu_{k-1})$ を選ぶ．

Step 2. **非単調手順**

Step 2.1 ニュートン方程式 (6)

$$J_k \Delta \boldsymbol{w}_k = -\boldsymbol{r}(\boldsymbol{w}_k; \mu_k)$$

を解くことによって探索方向 $\Delta \boldsymbol{w}_k$ を計算する．

Step 2.2 $\boldsymbol{\xi}_k + \alpha\Delta\boldsymbol{\xi}_k > 0$，$\boldsymbol{z}_k + \alpha\Delta\boldsymbol{z}_k > 0$ を満たすように直線探索を行い，ステップ幅 α を求める．

Step 2.3 もし $F(\boldsymbol{x}_k + \alpha\Delta\boldsymbol{x}_k, \mu_k) \ge \lambda_k$ ならば $\lambda_{k+1} = \lambda_k$ とおいて，Step3 へ行く．

Step 2.4 $\lambda_{k+1} \in [\max\{F(\boldsymbol{x}_k, \mu_k), F(\boldsymbol{x}_k + \alpha\Delta\boldsymbol{x}_k, \mu_k)\}, F(\boldsymbol{x}_0, \mu_{-1})]$ とおく．

Step 3. もし $\|\boldsymbol{r}(\boldsymbol{w}_k + \alpha\Delta\boldsymbol{w}_k, \mu_k)\| \le M_c\mu_k$ ならば $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \alpha\Delta\boldsymbol{w}_k$ とおく．

Step 4. $k = k + 1$ とおいて，Step1 へいく．

2　制約付き大規模最適化問題に対するアルゴリズム

変数の数が非常に大きい問題，いわゆる大規模問題（Large scale problem）に対する数値解法が近年強く望まれている．無制約最小化問題については，いくつかの解法が開発されており，非厳密 Newton 法や共役勾配法などが良く知られている．一方，準 Newton 法は，そうした解法に比べて近似行列を保存する点において不利である．しかしながら，ヘッセ行列はスパース性や分割性などの特別な構造を持っていることが多いので，近似行列 $\boldsymbol{B}_k$ にこうした特別な構造を組み込んだ準 Newton 法が提案されている．大規模問題を考慮した代表的な準 Newton 法として，次の 3 つが挙げられる．

1. スパース準 Newton 法
2. 分割準 Newton 法
3. 記憶制限付き準 Newton 法

スパース準 Newton 法と分割準 Newton 法は，ヘッセ行列の特別な構造を利用した解法である．それに対して，ヘッセ行列の特別な構造を利用するわけではないが，近似行列の更新公式を行列の形ではなく，数本のベクトルを用いて保存することによって計算機の記憶容量を大幅に削減した準 Newton 法が考えられている．これが，2.1 節で述べる**記憶制限付き準 Newton 法**（limited memory quasi-Newton method）である．

2.1　Nocedal の記憶制限準 Newton 法

記憶制限付き準 Newton 法は，共役勾配法に関連して，大規模な無制約最小化問題を解くために Nocedal[4] によって提案された方法である．

基本的な考え方は，ヘッセ行列の逆行列 $\nabla^2 f(\boldsymbol{x}_k)^{-1}$ を近似するのに H 公式の BFGS 公式を用いる． k 回目の反復における H 公式の BFGS 公式は

$$\begin{aligned} H_k &= H_{k-1} - \frac{H_{k-1}\boldsymbol{y}_{k-1}{\boldsymbol{s}_{k-1}}^T + \boldsymbol{s}_{k-1}{\boldsymbol{y}_{k-1}}^T H_{k-1}}{{\boldsymbol{s}_{k-1}}^T\boldsymbol{y}_{k-1}} \\ &\qquad + \left(1 + \frac{{\boldsymbol{y}_{k-1}}^T H_{k-1}\boldsymbol{y}_{k-1}}{{\boldsymbol{s}_{k-1}}^T\boldsymbol{y}_{k-1}}\right)\frac{\boldsymbol{s}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T\boldsymbol{y}_{k-1}} \\ &= \left(I - \frac{\boldsymbol{y}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T\boldsymbol{y}_{k-1}}\right)^T H_{k-1}\left(I - \frac{\boldsymbol{y}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T\boldsymbol{y}_{k-1}}\right) + \frac{\boldsymbol{s}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T\boldsymbol{y}_{k-1}} \end{aligned}$$

と書けるので，

$$V_{k-1} = I - \frac{\boldsymbol{y}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T\boldsymbol{y}_{k-1}}$$

とおけば

$$H_k = {V_{k-1}}^T H_{k-1} V_{k-1} + \frac{\boldsymbol{s}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T\boldsymbol{y}_{k-1}}$$

を得る．このとき $(k-1)$ 回目の BFGS 公式

$$H_{k-1} = {V_{k-2}}^T H_{k-2} V_{k-2} + \frac{\boldsymbol{s}_{k-2}{\boldsymbol{s}_{k-2}}^T}{{\boldsymbol{s}_{k-2}}^T \boldsymbol{y}_{k-2}}$$

を上式に代入すれば

$$H_k = {V_{k-1}}^T {V_{k-2}}^T H_{k-2} V_{k-2} V_{k-1} + {V_{k-1}}^T \frac{\boldsymbol{s}_{k-2}{\boldsymbol{s}_{k-2}}^T}{{\boldsymbol{s}_{k-2}}^T \boldsymbol{y}_{k-2}} V_{k-1} + \frac{\boldsymbol{s}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T \boldsymbol{y}_{k-1}}$$

となり，この操作を順々に繰り返して番号を遡っていけば，結局

$$\begin{aligned}
H_k \;=\; & (V_0 V_1 \cdots V_{k-2} V_{k-1})^T H_0 (V_0 V_1 \cdots V_{k-2} V_{k-1}) \\
& + (V_1 \cdots V_{k-2} V_{k-1})^T \frac{\boldsymbol{s}_0 {\boldsymbol{s}_0}^T}{{\boldsymbol{s}_0}^T \boldsymbol{y}_0} (V_1 \cdots V_{k-2} V_{k-1}) \\
& + \cdots \\
& + (V_{k-2} V_{k-1})^T \frac{\boldsymbol{s}_{k-3}{\boldsymbol{s}_{k-3}}^T}{{\boldsymbol{s}_{k-3}}^T \boldsymbol{y}_{k-3}} (V_{k-2} V_{k-1}) \\
& + {V_{k-1}}^T \frac{\boldsymbol{s}_{k-2}{\boldsymbol{s}_{k-2}}^T}{{\boldsymbol{s}_{k-2}}^T \boldsymbol{y}_{k-2}} V_{k-1} \\
& + \frac{\boldsymbol{s}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T \boldsymbol{y}_{k-1}}
\end{aligned} \tag{9}$$

を得る．ここで， H_0 は与えられた正定値対称行列かつ対角な初期行列である．このままでは従来の準 Newton 法と本質的には変わらないが，Nocedal は $V_0 V_1 \cdots V_k$ をすべて保存するのではなくて， k 回目の反復で過去 t 個分だけを保存して行列更新することを提案した．すなわち

$$\begin{aligned}
H_k \;=\; & (V_{k-t} V_{k-t+1} \cdots V_{k-2} V_{k-1})^T (V_{k-t} V_{k-t+1} \cdots V_{k-2} V_{k-1}) \\
& + (V_{k-t+1} V_{k-t+2} \cdots V_{k-2} V_{k-1})^T \frac{\boldsymbol{s}_{k-t}{\boldsymbol{s}_{k-t}}^T}{{\boldsymbol{s}_{k-t}}^T \boldsymbol{y}_{k-t}} (V_{k-t+1} V_{k-t+2} \cdots V_{k-2} V_{k-1}) \\
& + \cdots \\
& + (V_{k-2} V_{k-1})^T \frac{\boldsymbol{s}_{k-3}{\boldsymbol{s}_{k-3}}^T}{{\boldsymbol{s}_{k-3}}^T \boldsymbol{y}_{k-3}} (V_{k-2} V_{k-1}) \\
& + {V_{k-1}}^T \frac{\boldsymbol{s}_{k-2}{\boldsymbol{s}_{k-2}}^T}{{\boldsymbol{s}_{k-2}}^T \boldsymbol{y}_{k-2}} V_{k-1} \\
& + \frac{\boldsymbol{s}_{k-1}{\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T \boldsymbol{y}_{k-1}}
\end{aligned} \tag{10}$$

として行列を更新することを提案した．ただし，$H_0 = I$ とおき，$k < t$ の時は式 (9) を用いる．過去の記録を制限して保存するという意味で，この方法を記憶制限付き準 Newton 法と呼び，特に上記の公式 (10) を**記憶制限付き BFGS 公式**（limited memory BFGS update）という．この公式 (10) では行列の形式を用いるのではなく，ベクトル $\boldsymbol{s}_i$， $\boldsymbol{y}_i$， $i = k - t, \cdots, k-1$ を保存しておけばよく，必要なときに随時ベクトルの内積計算等を実行して探索方向 $\boldsymbol{d}_k$ を求める．このことによって，大規模な NLP 問題に準 Newton 法を適用することが可能になる．実際の計算では過去の履歴の個数 t はせいぜい 5〜10 程度に設定されるので，計算量の大幅な節約が実現できる．

2.2 Powell の修正 BFGS 公式

本研究のアルゴリズムでは，$B_k^{-1}\,(=H_k)$ の更新公式が必要である．したがって，$B_k^{-1}\,(=H_k)$ の更新公式は H 公式の BFGS 公式を用いる．

【H 公式の BFGS 公式】

$$H_{k+1} \;=\; H_k - \frac{H_k\boldsymbol{y}_k\boldsymbol{s}_k^T + \boldsymbol{s}_k\boldsymbol{y}_k^T H_k}{\boldsymbol{s}_k^T\boldsymbol{y}_k} + \left(1 + \frac{\boldsymbol{y}_k^T H_k \boldsymbol{y}_k}{\boldsymbol{s}_k^T\boldsymbol{y}_k}\right)\frac{\boldsymbol{s}_k\boldsymbol{s}_k^T}{\boldsymbol{s}_k^T\boldsymbol{y}_k}$$

しかし，ラグランジュ関数のヘッセ行列を近似した場合，$\boldsymbol{s}_k^T\boldsymbol{y}_k > 0$ とは限らず，H_k の正定値性の保存が保証されていないため，次のような修正を加えた公式を用いる．

【H 公式の修正 BFGS 公式】

$$H_{k+1} \;=\; H_k - \frac{H_k\boldsymbol{z}_k\boldsymbol{s}_k^T + \boldsymbol{s}_k\boldsymbol{z}_k^T H_k}{\boldsymbol{s}_k^T\boldsymbol{z}_k} + \left(1 + \frac{\boldsymbol{z}_k^T H_k \boldsymbol{z}_k}{\boldsymbol{s}_k^T\boldsymbol{z}_k}\right)\frac{\boldsymbol{s}_k\boldsymbol{s}_k^T}{\boldsymbol{s}_k^T\boldsymbol{z}_k}$$

ただし，

$$\begin{aligned}
\boldsymbol{z}_k &= \psi_k\boldsymbol{s}_k + (1-\psi_k)\,H_k\boldsymbol{y}_k \\
\psi_k &= \begin{cases} 1 & \boldsymbol{s}_k^T\boldsymbol{y}_k \ge \omega\boldsymbol{s}_k^T H_k\boldsymbol{y}_k \text{のとき} \\ \frac{(1-\omega)\boldsymbol{y}_k^T H_k\boldsymbol{y}_k}{\boldsymbol{y}_k^T(H_k\boldsymbol{y}_k - \boldsymbol{s}_k)} & \text{上記以外の時} \end{cases}
\end{aligned}$$

パラメータ ω は $\omega = 0.1$ または $\omega = 0.2$ とおかれる．この公式は，セカント条件を緩和した条件

$$B_{k+1}\boldsymbol{s}_k = \boldsymbol{z}_k$$

を満足する．また，常に $\boldsymbol{s}_k^T\boldsymbol{z}_k > 0$ が成り立つので，上記の公式は正定値性を保存する．この更新公式を Powell の修正 BFGS 公式という．

2.3 探索方向の計算手順

ヘッセ行列の逆行列 $\nabla^2 f(x_k)^{-1}$ を行列 H で近似したニュートン方程式 (6) を解くと，

$$\begin{pmatrix} \Delta\boldsymbol{x}_k \\ \Delta\boldsymbol{y}_k \\ \Delta\boldsymbol{z}_k \end{pmatrix} = -\begin{pmatrix} H_k^{-1} & -\nabla\boldsymbol{h}(\boldsymbol{x}_k) & -D^T \\ \nabla\boldsymbol{h}(\boldsymbol{x}_k)^T & \boldsymbol{0} & \boldsymbol{0} \\ Z_k D & \boldsymbol{0} & DX_kD^T \end{pmatrix}^{-1}\begin{pmatrix} \nabla_{\boldsymbol{x}}L(\boldsymbol{w}_k) \\ \boldsymbol{h}(\boldsymbol{x}_k) \\ DX_kD^TZ_k\boldsymbol{e} - \mu\boldsymbol{e} \end{pmatrix} \tag{11}$$

となる．ここで，

$$\begin{aligned}
J &= \begin{pmatrix} \bar{J} & K \\ W^T & DXD^T \end{pmatrix}, \bar{J} = \begin{pmatrix} H^{-1} & -\nabla\boldsymbol{h} \\ \nabla\boldsymbol{h}^T & \boldsymbol{0} \end{pmatrix}, \\
K &= \begin{pmatrix} -D^T \\ \boldsymbol{0} \end{pmatrix}, W = \begin{pmatrix} D^TZ \\ \boldsymbol{0} \end{pmatrix}, \\
\Delta\boldsymbol{P} &= \begin{pmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{y} \end{pmatrix}, \boldsymbol{q} = \begin{pmatrix} \nabla L \\ \boldsymbol{h} \end{pmatrix}, \boldsymbol{\gamma} = DXD^TZ\boldsymbol{e} - \mu\boldsymbol{e}
\end{aligned}$$

とおくと式 (11) は次のようになる．

$$\begin{pmatrix} \Delta \boldsymbol{P} \\ \Delta \boldsymbol{z} \end{pmatrix} = -\begin{pmatrix} \bar{J} & K \\ W^T & DXD^T \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{q} \\ \boldsymbol{\gamma} \end{pmatrix}.$$

ここで，J^{-1} と $\bar{J}^{-1}$ は解析的に次のように求められる．

$$J^{-1} = \begin{pmatrix} \bar{J}^{-1}(I + KQ^{-1}W^T\bar{J}^{-1}) & -\bar{J}^{-1}KQ^{-1} \\ -Q^{-1}W^T\bar{J}^{-1} & Q^{-1} \end{pmatrix}, \qquad Q = DXD^T - W^T\bar{J}^{-1}K$$

$$\bar{J}^{-1} = \begin{pmatrix} H - H\nabla \boldsymbol{h}U\nabla \boldsymbol{h}^T H & H\nabla \boldsymbol{h}U \\ -U\nabla \boldsymbol{h}^T H & U \end{pmatrix}, \qquad U = (\nabla \boldsymbol{h}^T H \nabla \boldsymbol{h})^{-1}$$

以上より，探索方向は

$$\begin{pmatrix} \Delta \boldsymbol{P} \\ \Delta \boldsymbol{z} \end{pmatrix} = -\begin{pmatrix} \bar{J}^{-1}\boldsymbol{q} + \bar{J}^{-1}K\Delta \boldsymbol{z} \\ \Delta \boldsymbol{z} \end{pmatrix}, \qquad \Delta \boldsymbol{z} = -Q^{-1}W^T\bar{J}^{-1}\boldsymbol{q} + Q^{-1}\boldsymbol{\gamma}$$

と求めることができる．これより，探索方向 $\Delta \boldsymbol{w}$ を求めるには，$\bar{J}^{-1}\boldsymbol{q}$，$\bar{J}^{-1}K\Delta \boldsymbol{z}$ を計算する必要がある．$\bar{J}^{-1}\boldsymbol{q}$，$\bar{J}^{-1}K\Delta \boldsymbol{z}$ を具体的に表すと

$$\bar{J}^{-1}\boldsymbol{q} = \begin{pmatrix} H\nabla L - H\nabla \boldsymbol{h}U\nabla \boldsymbol{h}^T H\nabla L + H\nabla \boldsymbol{h}U\boldsymbol{h} \\ -U\nabla \boldsymbol{h}^T H\nabla L + U\boldsymbol{h} \end{pmatrix} \tag{12}$$

$$\bar{J}^{-1}K\Delta \boldsymbol{z} = \begin{pmatrix} -HD^T\Delta \boldsymbol{z} + H\nabla \boldsymbol{h}U\nabla \boldsymbol{h}^T HD^T\Delta \boldsymbol{z} \\ U\nabla \boldsymbol{h}^T HD^T\Delta \boldsymbol{z} \end{pmatrix} \tag{13}$$

となる．ここで注意したいのは式 (12)，(13) の計算で出てくる $H\nabla L$，$H\nabla \boldsymbol{h}$，$HD^T\Delta \boldsymbol{z}$ の演算で H_k を保存すると大規模問題では非常に大きな記憶領域が必要となり，計算機の限界等により計算を行うことが困難となる．そこで，本研究では次の様に演算することによりメモリの節約を行うことを提案する．なお，今回対象としている問題は等式制約の数 l が小さいので $H\nabla L$，$H\nabla \boldsymbol{h}$，$HD^T\Delta \boldsymbol{z}$ の演算は H と n 次元ベクトルの内積とみなすことができる．したがって以下では $H\nabla L$ についてだけ説明し他の 2 つは省略する．
Limited Memory BFGS 法 (2.2 節) を用いることにより，行列 $H\nabla L$ は次のように表すことができる．

$$\begin{aligned} H_k\nabla L(\boldsymbol{x}_k) = \; & (V_{k-t}V_{k-t+1}\cdots V_{k-2}V_{k-1})^T (V_{k-t}V_{k-t+1}\cdots V_{k-2}V_{k-1})\nabla L(\boldsymbol{x}_k) \\ & + (V_{k-t+1}\cdots V_{k-2}V_{k-1})^T \frac{\boldsymbol{s}_{k-t}\boldsymbol{s}_{k-t}{}^T}{\boldsymbol{s}_{k-t}{}^T\boldsymbol{z}_{k-t}} (V_{k-t+1}\cdots V_{k-2}V_{k-1})\nabla L(\boldsymbol{x}_k) \\ & + \quad \cdots \\ & + (V_{k-1}V_{k-2})^T \frac{\boldsymbol{s}_{k-3}\boldsymbol{s}_{k-3}{}^T}{\boldsymbol{s}_{k-3}{}^T\boldsymbol{z}_{k-3}} (V_{k-1}V_{k-2})\nabla L(\boldsymbol{x}_k) \\ & + V_{k-1}^T \frac{\boldsymbol{s}_{k-2}\boldsymbol{s}_{k-2}{}^T}{\boldsymbol{s}_{k-2}{}^T\boldsymbol{z}_{k-2}} V_{k-1}\nabla L(\boldsymbol{x}_k) \\ & + \frac{\boldsymbol{s}_{k-1}\boldsymbol{s}_{k-1}{}^T}{\boldsymbol{s}_{k-1}{}^T\boldsymbol{z}_{k-1}} \nabla L(\boldsymbol{x}_k) \end{aligned} \tag{14}$$

このように数本のベクトル演算の形に表される．しかし，このまま行列 $H_k \nabla L(\boldsymbol{x}_k)$ を求める場合，行列 $V_{k-1}, \cdots, V_1$ を保存しておかなければならず，非効率であると言える．したがって，上記のように表された行列 $H_k \nabla L(\boldsymbol{x}_k)$ の効率的な計算の提案を行う．

行列 $H \nabla L$ の式は基本的に次のような構造をしている．

$$\boxed{(V_{k-t+1} \cdots V_{k-1})^T \frac{\boldsymbol{s}_{k-t} {\boldsymbol{s}_{k-t}}^T}{{\boldsymbol{s}_{k-t}}^T \boldsymbol{z}_{k-t}} (V_{k-t+1} \cdots V_{k-1}) \nabla L(\boldsymbol{x}_k)}$$

ここで，行列 D_i を次のような形で定義する．

$$\begin{aligned} D_i &= V_i \cdots V_{k-1} \nabla L(\boldsymbol{x}_k) \\ & \left(\text{ただし}, V_i = I - \frac{\boldsymbol{z}_i {\boldsymbol{s}_i}^T}{{\boldsymbol{s}_i}^T \boldsymbol{z}_i} \right) \end{aligned} \tag{15}$$

式 (15) をそのまま計算すると $V_i \cdots V_{k-1}$ を全て保存しておかなければならない．そこで効率的な計算として，次の様におこなう．

$$D_i = V_i \quad V_{i-1} \quad \underline{\cdots \quad \underline{V_{k-2} \quad \underline{V_{k-1} \quad \nabla L(\boldsymbol{x}_k)}}} \tag{16}$$

まず，式 (16) の下線部分　の計算をおこなう．

$$D_i^{[0]} = \nabla L(\boldsymbol{x}_k) \text{ とおく}$$

$$\begin{aligned} D_i^{[1]} &= V_{k-1} \nabla L(\boldsymbol{x}_k) \\ &= V_{k-1} D_i^{[0]} \\ &= \left(I - \frac{\boldsymbol{z}_{k-1} {\boldsymbol{s}_{k-1}}^T}{{\boldsymbol{s}_{k-1}}^T \boldsymbol{z}_{k-1}} \right) D_i^{[0]} \end{aligned} \tag{17}$$

次に式 (17) を用いて，式 (16) の下線部分　の計算を行う．

$$\begin{aligned} D_i^{[2]} &= V_{k-2} V_{k-1} \nabla L(\boldsymbol{x}_k) \\ &= V_{k-2} D_i^{[1]} \\ &= \left(I - \frac{\boldsymbol{z}_{k-2} {\boldsymbol{s}_{k-2}}^T}{{\boldsymbol{s}_{k-2}}^T \boldsymbol{z}_{k-2}} \right) D_i^{[1]} \end{aligned} \tag{18}$$

次に式 (18) を用いて，式 (16) の下線部分　の計算を行う．

$$\begin{aligned} D_i^{[3]} &= V_{k-3} V_{k-2} V_{k-1} \nabla L(\boldsymbol{x}_k) \\ &= V_{k-3} D_i^{[2]} \\ &= \left(I - \frac{\boldsymbol{z}_{k-3} {\boldsymbol{s}_{k-3}}^T}{{\boldsymbol{s}_{k-3}}^T \boldsymbol{z}_{k-3}} \right) D_i^{[2]} \end{aligned}$$

このように右から順々に計算していくことにより，すべての行列領域 $V_i \cdots V_{k-1}$ を保存しておく必要はなく，ベクトル演算の形で求めることができる．
上記の計算をまとめると次のようなアルゴリズムになる．

【D_i の計算アルゴリズム】

Step 0. $D_i = \nabla L(\boldsymbol{x}_k)$ とおく．

Step 1. $l = (k-1) \to i$ まで下の計算を行う．

$$D_i = \left(I - \frac{\boldsymbol{z}_l \boldsymbol{s}_l{}^T}{\boldsymbol{s}_l{}^T \boldsymbol{z}_l} \right) D_i$$

このような計算方法を用いることにより，効率的な計算を行うことができる．

2.4 拡張セカント条件を満たす公式

本節では，拡張セカント条件を満たすという性質を損なうことなくこの公式を簡略化し，記憶領域を低減しつつ探索方向を計算できる新しい公式を用いた記憶制限準ニュートン法について述べる．また，拡張されたセカント条件を満たす更新公式は，サイジング因子 w_k を用いることにより，正値対称行列 G を持つ線形方程式 $f(\boldsymbol{x}) = G\boldsymbol{x} - \boldsymbol{b}$ に対して，H_k の固有値を G^{-1} の固有値に単調に近づけることが出来る．このような考え方にもとづいて，根岸ら [3] は大規模非線形方程式の解法に用い成功している．

近似行列 H_k が満たす重要な条件がセカント条件である．$\nabla^2 f(\boldsymbol{x}_k)$ の近似行列を B_k，その逆行列を H_k としたときに，$\nabla f(\boldsymbol{x})$ のテイラー展開

$$\nabla f(\boldsymbol{x}_k) = \nabla f(\boldsymbol{x}_{k+1}) + \nabla^2 f(\boldsymbol{x}_{k+1})(\boldsymbol{x}_k - \boldsymbol{x}_{k+1}) + \cdots$$

を $(\boldsymbol{x}_k - \boldsymbol{x}_{k+1})$ の項で打ち切り，

$$\boldsymbol{s}_k = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k, \quad \boldsymbol{y}_k = \nabla f(\boldsymbol{x}_{k+1}) - \nabla f(\boldsymbol{x}_k) \tag{19}$$

とすると，$\nabla^2 f(\boldsymbol{x}_{k+1})\boldsymbol{s}_k \simeq \boldsymbol{y}_k$ なので，近似行列 B_{k+1} が

$$B_{k+1}\boldsymbol{s}_k = \boldsymbol{y}_k \tag{20}$$

を満たすことが要請される．これは $\boldsymbol{s}_k$ 方向で，$\nabla f(\boldsymbol{x}_{k+1})$ を近似することを課すもので，セカント条件と呼ばれている．H 公式ではセカント条件は

$$\boldsymbol{s}_k = H_{k+1}\boldsymbol{y}_k \tag{21}$$

である．ここで

$$\begin{aligned} S_{k+1} &= [\boldsymbol{s}_0, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_k], \\ Y_{k+1} &= [\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k] \end{aligned} \tag{22}$$

と定め，拡張されたセカント条件とは

$$S_{k+1} = H_{k+1} Y_{k+1} \tag{23}$$

である [5]．これはすなわち H_{k+1} が $\boldsymbol{s}_0, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_k$ と $\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k$ に対してセカント条件を満たしているということである．Yamaki and Yabe[7],Yabe and Yamaki[5] は，近似行列 H_{k+1} の生成において，拡張セカント条件を満たすサイジング付公式を提案している．サイジングとは近似行列 H_k をそのまま更新するのではなく, 適当な正の数 w_k を H_k にかけてから更新することである．Yabe and Yamaki[5] では，w_k の選び方として，

$$
\begin{aligned}
w_0 &= \frac{(1-\psi_0)\boldsymbol{s}_0^T\boldsymbol{y}_0}{\boldsymbol{y}_0^T H_0 \boldsymbol{y}_0} + \frac{\psi_0 \boldsymbol{s}_0^T \nabla f(\boldsymbol{x}_0)}{\nabla f(\boldsymbol{x}_0)^T H_0 \boldsymbol{y}_0}, \quad \psi_0 \in [0,1] \\
w_k &= \frac{\psi_k^1 \boldsymbol{s}_k^T \boldsymbol{y}_k}{\boldsymbol{y}_k^T H_k \boldsymbol{y}_k} + \frac{\psi_k^2 \boldsymbol{s}_k^T \nabla f(\boldsymbol{x}_k)}{\nabla f(\boldsymbol{x}_k)^T H_k \boldsymbol{y}_k} + \psi_k^3 \\
& \sum_{i=1}^{3} \psi_k^i = 1, \quad \psi_k^1, \psi_k^2, \psi_k^3 \geq 0, \quad \text{for} \quad k \geq 0
\end{aligned}
$$

を提案している．したがって，$k=0$ の場合には $H_0 = \mathrm{I}$ なので，簡単な計算によって w_0 が得られ，$k>0$ の場合は $w_k = 1$ でよいことがわかる．

公式は以下のとおりである．

$$
\begin{cases}
H_k = P_k + R_k \\
P_k = w_{k-1} L_k^T P_{k-1} L_k \\
L_k = I - Y_k \left(S_k^T Y_k\right)^{-1} S_k^T \\
R_k = S_k \left(S_k^T Y_k\right)^{-1} S_k^T \\
\quad H_i, P_i, R_i \in R^{n\times n}, \quad S_i, Y_i \in R^{n\times i}
\end{cases}
\tag{24}
$$

2.5 拡張セカント条件を満たす公式の記憶制限準ニュートン法

式 (24) の P_{k-1} に P_{k-2} から P_0 を逐次代入すると以下の式になる．

$$
\begin{aligned}
P_k &= w_{k-1} L_k^T P_{k-1} L_k \\
&= w_{k-1} \cdots w_0 L_k^T \cdots L_1^T P_0 L_1 \cdots L_k
\end{aligned}
$$

ここで，$L_1 \cdots L_k$ を使って計算するのではなく，$L_{k-t+1} \cdots L_k$ を使って近似的に P_k を計算する．すなわち

$$
P_k \simeq w_{k-1} \cdots w_{k-t} L_k^T \cdots L_{k-t+1}^T P_{k-t} L_{k-t+1} \cdots L_k
$$

とする．ここでは，$w_{k-1}, \ldots, w_{k-t+1} = 1$ でよく，また，$P_{k-t} = I$ とおいて，L_k だけ用いて計算すると，

$$
P_k \simeq w_{k-t} L_k^T L_k \tag{25}
$$

となる．従って H_k は

$$
\begin{aligned}
H_k &= P_k + R_k \\
&\simeq w_{k-t} L_k^T L_k + R_k
\end{aligned}
\tag{26}
$$

である．

探索方向 $\boldsymbol{d}_k$ は，

$$\begin{aligned}\boldsymbol{d}_k &= -H_k \nabla f(\boldsymbol{x}_k) \\ &= (P_k + R_k)\nabla f(\boldsymbol{x}_k)\end{aligned} \tag{27}$$

であり，

$$\begin{aligned}(P_k + R_k)\nabla f(\boldsymbol{x}_k) &\simeq w_{k-t} L_k^T L_k \nabla f(\boldsymbol{x}_k) + R_k \nabla f(x_k) \\ &= w_{k-t}(I - S_k(Y_k^T S_k)^{-1} Y_k^T)(I - Y_k(S_k^T Y_k)^{-1} S_k^T)\nabla f(\boldsymbol{x}_k) \\ &\quad + S_k(S_k^T Y_k)^{-1} S_k^T \nabla f(x_k)\end{aligned}$$

である．従って $(P_k{+}R_k)\nabla f(\boldsymbol{x}_k)$ を記憶領域を低減しつつ計算できればよい．まず，$R_k \nabla f(x_k)$ の計算であるが，$R_k \nabla f(x_k)$ の構造を以下に示す．

$$\begin{aligned}R_k \nabla f(\boldsymbol{x}_k) &= S_k(S_k^T Y_k)^{-1} S_k^T \nabla f(\boldsymbol{x}_k) \\ &= [\boldsymbol{s}_{k-t} \ldots \boldsymbol{s}_{k-1}] \left(\begin{bmatrix} \boldsymbol{s}_{k-t}^T \\ \vdots \\ \boldsymbol{s}_{k-1}^T \end{bmatrix} [\boldsymbol{y}_{k-t} \cdots \boldsymbol{y}_{k-1}] \right)^{-1} \begin{bmatrix} \boldsymbol{s}_{k-t}^T \\ \vdots \\ \boldsymbol{s}_{k-1}^T \end{bmatrix} \nabla f(\boldsymbol{x}_k)\end{aligned}$$

$$\boldsymbol{a} = \left(\begin{bmatrix} \boldsymbol{s}_{k-t}^T \\ \vdots \\ \boldsymbol{s}_{k-1}^T \end{bmatrix} [\boldsymbol{y}_{k-t} \cdots \boldsymbol{y}_{k-1}] \right)^{-1} \begin{bmatrix} \boldsymbol{s}_{k-t}^T \\ \vdots \\ \boldsymbol{s}_{k-1}^T \end{bmatrix} \nabla f(\boldsymbol{x}_k)$$

$$R_k \nabla f(\boldsymbol{x}_k) = S_k \boldsymbol{a}, \quad \boldsymbol{a} \in R^t$$

$R_k \nabla f(\boldsymbol{x}_k)$ において，$(S_k^T Y_k)^{-1} S_k^T \nabla f(\boldsymbol{x}_k)$ の計算の途中では，$t \times t$ の行列ないし，n 次元ベクトルが生成されるだけである．従って，この部分をベクトル $\boldsymbol{a}$ とおくと，$\boldsymbol{a}$ を先に計算して $S_k \boldsymbol{a}$ を後から計算することによって，$R_k \nabla f(\boldsymbol{x}_k)$ を記憶領域を低減しつつ計算できる．$P_k \nabla f(\boldsymbol{x}_k)$ についても同様に計算できる．従って，通常の準ニュートン法が $n \times n$ の行列を記憶しなければならないのに比べて，記憶制限準ニュートン法は数本のベクトルだけを記憶すればいいので，大幅に記憶領域を低減できる．

2.6 $(S_k^T Y_k)^{-1}$ の計算

本研究では，$(S_k^T Y_k)^{-1}$ の計算が必要である．しかし，このとき $S_k^T Y_k$ の正則性は常に保証されるとは限らない．そこで，正則性をこわす $\boldsymbol{s}_k$,$\boldsymbol{y}_k$ の組を除外しながら逆行列を計算することを考える．$(S_k^T Y_k)^{-1}$ を計算するために次の補題を使う．

補題 1: D が正則でかつ $a - \boldsymbol{b}^T D^{-1} \boldsymbol{c} \neq 0$ のとき，

$$\begin{pmatrix} a & \boldsymbol{b}^T \\ \boldsymbol{c} & D \end{pmatrix}^{-1} = \begin{pmatrix} r & -r\boldsymbol{b}^T D^{-1} \\ -rD^{-1}\boldsymbol{c} & D^{-1} + rD^{-1}\boldsymbol{c}\boldsymbol{b}^T D^{-1} \end{pmatrix}$$

ただし，$r = 1/(a - \boldsymbol{b}^T D^{-1} \boldsymbol{c})$.

実際に $(S_k^T Y_k)^{-1}$ は次のような手順で計算する．

$(S_k^T Y_k)^{-1}$ **の計算手順**
(1)　$A_1 = (\boldsymbol{s}_{k-1}^T \boldsymbol{y}_{k-1})$ とおく．
(2)　以下の手順を $i = 2$ から t まで繰り返す．

$$A_i = \begin{pmatrix} \boldsymbol{s}_{k-i}^T \boldsymbol{y}_{k-i} & \boldsymbol{s}_{k-i}^T B_i \\ C_i^T \boldsymbol{y}_{k-i} & A_{i-1} \end{pmatrix}$$

ただし，$B_i = [\boldsymbol{y}_{k-i+1}, \cdots, \boldsymbol{y}_{k-1}], C_i = [\boldsymbol{s}_{k-i+1}, \cdots, \boldsymbol{s}_{k-1}]$.
ここで，$\boldsymbol{s}_{k-i}^T \boldsymbol{y}_{k-i} - \boldsymbol{s}_{k-i}^T B_i (A_{i-1})^{-1} C_i^T \boldsymbol{y}_{k-i} = 0$ ならば，$S_k^T Y_k$ の正則性が保証されないので $\boldsymbol{s}_{k-i}, \boldsymbol{y}_{k-i}$ を除外し $A_i = A_{i-1}$ とおく．
そして $\boldsymbol{s}_{k-i}^T \boldsymbol{y}_{k-i} - \boldsymbol{s}_{k-i}^T B_i (A_{i-1})^{-1} C_i^T \boldsymbol{y}_{k-i} \neq 0$ のとき補題にしたがって以下のように $(S_k^T Y_k)^{-1}$ を計算する．

$$(A_i)^{-1} = \begin{pmatrix} r & -r \boldsymbol{s}_{k-i}^T B_i (A_{i-1})^{-1} \\ -r (A_{i-1})^{-1} C_i^T \boldsymbol{y}_{k-i} & (A_{i-1})^{-1} + r (A_{i-1})^{-1} C_i^T \boldsymbol{y}_{k-i} \boldsymbol{s}_{k-i}^T B_i (A_{i-1})^{-1} \end{pmatrix}$$

ただし，$r = (\boldsymbol{s}_{k-i}^T \boldsymbol{y}_{k-i} - \boldsymbol{s}_{k-i}^T B_i (A_{i-1})^{-1} C_i^T \boldsymbol{y}_{k-i})^{-1}$
$(A_t)^{-1}$ が求める $(S_k^T Y_k)^{-1}$ となる．
なお，正則性を崩す $\boldsymbol{s}_{k-i}$,$\boldsymbol{y}_{k-i}$ の組が除外された場合，$\boldsymbol{s}_{k-i}$,$\boldsymbol{y}_{k-i}$ を保存しておくメモリから除外されるため，実際の過去の履歴の保存数はそれぞれ $t-1$ 本ずつとなる．

2.7　信頼領域法 [9]

2.2 節と同様に，拡張セカント条件を満たす公式を用いラグランジュ関数のヘッセ行列を近似した場合，$\boldsymbol{s}_k^T \boldsymbol{y}_k > 0$ とは限らず，H_k の正定値性の保存が保証されていない．したがって大域的収束性を期待し，探索方向は信頼領域法で求める．以下では，バリヤ KKT 条件を満たす点を見つけるための信頼領域法について説明する．

2.7.1　準備

1.1 節で述べたメリット関数に対して，$\boldsymbol{x} \in \boldsymbol{R}^n$ で $\boldsymbol{s} \in \boldsymbol{R}^n$ についての 1 次近似関数 $F_l(\boldsymbol{x}, \boldsymbol{s}) : \boldsymbol{R}_+^n \to \boldsymbol{R}$ は次のように定義される．

$$F_l(\boldsymbol{x}, \boldsymbol{s}) = F(\boldsymbol{x}, \mu) + (\nabla f(\boldsymbol{x}) - \mu \Xi^{-1} \boldsymbol{e})^T \boldsymbol{s} + \rho \sum_{i=1}^{m} \Big(|h_i(\boldsymbol{x}) + \nabla h_i(\boldsymbol{x})^T \boldsymbol{s}| - |h_i(\boldsymbol{x})| \Big) \quad (28)$$

同様に，2 次近似関数 $F_q(\boldsymbol{x}, \boldsymbol{s}) : \boldsymbol{R}_+^n \to \boldsymbol{R}$ は次のようになる．

$$F_q(\boldsymbol{x}, \boldsymbol{s}) = F_l(\boldsymbol{x}, \mu) + \frac{1}{2} \boldsymbol{s}^T Q \boldsymbol{s} \quad (29)$$

ここで $Q \in \boldsymbol{R}^{n \times n}$ は任意の行列である．しかしながら，速い収束を実現するために次のようにおく．

$$Q = H + D^T \Xi^{-1} Z D \quad (30)$$

また，ステップ $\boldsymbol{s}$ でのこれらの関数の変化量は次のようになる．

$$\begin{aligned}
\Delta F_l(\boldsymbol{x},\boldsymbol{s}) &\equiv F_l(\boldsymbol{x},\boldsymbol{s}) - F_l(\boldsymbol{x},\boldsymbol{0}) = F_l(\boldsymbol{x},\boldsymbol{s}) - F(\boldsymbol{x},\mu)\\
\Delta F_q(\boldsymbol{x},\boldsymbol{s}) &\equiv F_q(\boldsymbol{x},\boldsymbol{s}) - F_q(\boldsymbol{x},\boldsymbol{0}) = F_q(\boldsymbol{x},\boldsymbol{s}) - F(\boldsymbol{x},\mu)\\
\Delta F(\boldsymbol{x},\boldsymbol{s}) &\equiv F(\boldsymbol{x}+\boldsymbol{s},\mu) - F(\boldsymbol{x},\mu)
\end{aligned}$$

2.7.2　探索方向

基準となる方向は，ニュートン方向 $\boldsymbol{\Delta w_N} = (\boldsymbol{\Delta x_N}, \boldsymbol{\Delta y_N}\boldsymbol{\Delta z_N})$ で次の方程式を解くことによって求まる．

$$\begin{pmatrix} B & -\nabla \boldsymbol{h}_(\boldsymbol{x}) & -D^T \\ \nabla \boldsymbol{h}(\boldsymbol{x})^T & \boldsymbol{0} & \boldsymbol{0} \\ ZD & \boldsymbol{0} & \Xi \end{pmatrix} \begin{pmatrix} \Delta \boldsymbol{x}_N \\ \Delta \boldsymbol{y}_N \\ \Delta \boldsymbol{z}_N \end{pmatrix} = -\boldsymbol{r}(\boldsymbol{w},\mu) \tag{31}$$

また，信頼領域法ではアルゴリズムに大域的収束性を含むために次の方程式を解いた方向も利用している．

$$\begin{pmatrix} C & -\nabla \boldsymbol{h}_(\boldsymbol{x})^T & -D^T \\ \nabla \boldsymbol{h}_(\boldsymbol{x})^T & \boldsymbol{0} & \boldsymbol{0} \\ ZD & \boldsymbol{0} & \Xi \end{pmatrix} \begin{pmatrix} \Delta \boldsymbol{x}_{SD} \\ \Delta \boldsymbol{y}_{SD} \\ \Delta \boldsymbol{z}_{SD} \end{pmatrix} = -\boldsymbol{r}(\boldsymbol{w},\mu) \tag{32}$$

ここで C は正定値対角行列であり，方向 $\Delta \boldsymbol{w}_{SD} = (\Delta \boldsymbol{x}_{SD}, \Delta \boldsymbol{y}_{SD}, \Delta \boldsymbol{z}_{SD})$ を最急降下方向と呼ぶことにする．

2.7.3　アルゴリズム

以下にパラメータ μ を固定したときのバリヤ KKT 点を求めるアルゴリズムを示す．k 回目の反復において信頼半径 $\delta_k > 0$，$\Delta \boldsymbol{w}_N$，$\Delta \boldsymbol{w}_{SD}$ を与える．これらの 2 つのベクトルから信頼領域制約 $\|\boldsymbol{s}_k\| \leq \delta_k$ と $\boldsymbol{\xi}_k \geq \boldsymbol{0}$，$\boldsymbol{z}_k \geq \boldsymbol{0}$ を満たす探索方向 $\boldsymbol{s}_k$ をつくる．

後半の条件を満たすために次の制約を満たすようにする．

$$(1-\gamma)(\boldsymbol{\xi}_k)_i \leq (D(\boldsymbol{x}_k + \boldsymbol{s}_k))_i, \qquad i = 1,\ldots,m$$

ここで $\gamma \in (0,1)$．この条件によって変数の非負制約を満たすために信頼半径を不必要に小さくする必要がなくなる．また，探索方向 $\boldsymbol{s}_k$ は次の条件を満たさなければならない．

$$\Delta F_q(\boldsymbol{x}_k, \boldsymbol{s}_k) \leq \frac{1}{2}\Delta F_q(\boldsymbol{x}_k, \alpha^*(\boldsymbol{x}_k, \Delta \boldsymbol{x}_{SDk})\Delta \boldsymbol{x}_{SDk}) \tag{33}$$

ここで $\alpha^*(\boldsymbol{x}, \boldsymbol{d})$ は次式で定義される．

$$\alpha^*(\boldsymbol{x}, \boldsymbol{d}) = \mathrm{argmin}\{F_q(\boldsymbol{x}, \alpha\boldsymbol{d})|\alpha \leq 1, \|\alpha\boldsymbol{d}\| \leq \delta, \alpha \in [0, \gamma\bar{\alpha}(\boldsymbol{x}, \boldsymbol{d})]\} \tag{34}$$

$$\bar{\alpha} = \min_i \{-\frac{\xi_i}{(Dd)_i}|(Dd)_i < 0\}.$$

条件 (33) は最急降下方向を基準とした場合の探索方向 $\boldsymbol{s}_k$ が十分な減少をする条件となる．

【アルゴリズム 信頼領域法】

Step 0. 初期点 $\boldsymbol{w}_0(\xi_0 > 0, z_0 > 0)$ と $\mu > 0$, $\rho > 0$, $\epsilon > 0$, $\gamma \in (0,1)$, $\delta_0 > 0$ を与え, $k = 0$ とおく.

Step 1. もし $\|\boldsymbol{r}(\boldsymbol{w}_k, \mu)\| \leq \epsilon$ ならば停止する.

Step 2. 式 (31), 式 (32) から $\Delta \boldsymbol{w}_{Nk}$, $\Delta \boldsymbol{w}_{SDk}$ をそれぞれ求める. もし

$$\|\Delta \boldsymbol{x}_{Nk}\| \leq M_1 \|\Delta \boldsymbol{x}_{SDk}\| \tag{35}$$

を満たしていなければ, (35) を満たすように B_k を緩和する. ただし $M_1 > 0$.

Step 3. 探索方向 $\bar{\boldsymbol{s}_k}$ を式 (36) により計算する.

$$\bar{\boldsymbol{s}_k} = \nu \Delta \boldsymbol{x}_{SDk} + (1-\nu)\Delta \boldsymbol{x}_{Nk} \tag{36}$$

ここで $\nu \in [0,1]$ は $\boldsymbol{s}_k = \alpha^*(\boldsymbol{x}_k, \bar{\boldsymbol{s}_k})\bar{\boldsymbol{s}_k}$ が条件 (37) を満たすように選ばれる.

$$\begin{array}{rcl} \|\boldsymbol{s}_k\| & \leq & \delta_k, \\ (1-\gamma)(\xi_k)_i & \leq & (D(x_k + s_k))_i, i = 1, \ldots, n, \\ \Delta F_q(\boldsymbol{x}_k, \boldsymbol{s}_k) & \leq & \dfrac{1}{2}\Delta F_q(\boldsymbol{x}_k, \alpha^*(\boldsymbol{x}_k, \Delta \boldsymbol{x}_{SDk})\Delta \boldsymbol{x}_{SDk}) \end{array} \tag{37}$$

Step 4. δ_{k+1} を次のように決める:

もし $\Delta F(\boldsymbol{x}_k, \boldsymbol{s}_k) > \dfrac{1}{4}\Delta F_q(\boldsymbol{x}_k, \boldsymbol{s}_k)$ ならば $\delta_{k+1} = \dfrac{1}{2}\delta_k$;

もし $\Delta F(\boldsymbol{x}_k, \boldsymbol{s}_k) \leq \dfrac{3}{4}\Delta F_q(\boldsymbol{x}_k, \boldsymbol{s}_k)$ ならば $\delta_{k+1} = 2\delta_k$;

それ以外は $\delta_{k+1} = \delta_k$.

Step 5. もし $\Delta F(\boldsymbol{x}_k, \boldsymbol{s}_k) \leq 0$ ならば, $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}_k$, $\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \Delta \boldsymbol{y}_k$, $\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \alpha_{zk}\Delta \boldsymbol{z}_k$ とする. さもなければ, $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k$ とする

Step 6. $k = k + 1$ とおいて, Step1 へいく.

注 1. Step1 の停止判定は $\epsilon = M_c\mu$ とし, 次の緩和されたバリヤ KKT 条件を用いる.

$$\|\boldsymbol{r}(\boldsymbol{w}_{k+1}, \mu_k)\| \leq M_c\mu_k$$

注 2. 信頼領域法によって生成される点列 $F(\boldsymbol{x}_k, \mu)$ は単調に減少する.

注 3. Step2 における緩和は, 対角行列を足すことによって行う.

注 4. Step3 では，$\nu \in [0,1]$ に対して $\boldsymbol{s}_k = \alpha^*(\boldsymbol{x}_k, \bar{\boldsymbol{s}_k})\bar{\boldsymbol{s}_k}$ を計算し，条件 (37) を満たしているか確認している．もし $\nu = 1$ ならば条件 (37) は明らかに満たされる．もし $\nu = 0$ ならば $\boldsymbol{s}_k$ はニュートン方向 $\Delta\boldsymbol{x}_{Nk}$ になる．したがって，最初に $\nu = 0$ とし条件 (37) を満たすまで ν の値を 0.1 ずつ増やしていく．この手順はそれぞれの $k = 0, 1, \ldots$ に対して

$$\|\boldsymbol{s}_k\| \leq M_2 \|\Delta\boldsymbol{x}_{SDk}\|$$

を満たす $M_2 > 0$ が存在することを保証する．

注 5. Step5 の α_{zk} の求め方は [9] を参照されたい．

3 考察

本研究では，大規模制約条件付非線形最適化問題に対する主双対内点法について議論した．大規模問題に対処するには，問題のラグランジュ関数のヘッセ行列の近似を記憶制限法による方法が有力である．ここで示した二つの方法には，それぞれ特徴がある．第一の方法は Nocedal の方法にもとづくもので，Powell の修正公式と併用することで，行列の正定値性を保証することができる．しかし，制約条件付問題の場合ラグランジュ関数のヘッセ行列は正定値性を持たない．また，計算手順が複雑である．

第二の方法は，拡張セカント条件を満たす公式である．この方法では正定値対称性が保証されないかわりに，二次関数に対する有限回収束が保証される．また，逐次探索方向の履歴の独立性を吟味する機構を備えることと，信頼領域法を組み合わせることで，性能の向上を目指している．

今後は，数値実験などにより性能評価を重ねて，より強固なアルゴリズムにしていくことが必要である．

参考文献

[1] S. P. Han, Superlinearly convergent variable metric algorithms for general non-1inear programming problems, *Mathematical Programming*, vol.11, pp.263-282, 1976.

[2] N. Maratos, Exact penalty function algorithms for finite dimensional and control optimization problems, *Ph.D.Thesis,Imperial College of Science and Technology,University of London*, 1978.

[3] 根岸　達彦，八巻　直一，矢部　博，大規模非線形方程式に対する準ニュートン法について，統計数理研究所共同研究リポート 161，「最適化：モデリングとアルゴリズム 16」, pp.1-8, 2003

[4] J. Nocedal, Updating quasi-Newton matrices with limited storage, *Mathematics and Computation*, vol.35, pp.773-782, 1980.

[5] H. Yabe and N. Yamaki, Some properties of Oren's SSVM type algorithm for unconstrained minimization, *TRU Mathematics*, vol.16-2, pp.103-111, 1980.

[6] 矢部　博, 八巻　直一, 非線形計画法, 朝倉書店, 1999.

[7] N. Yamaki and H. Yabe, A Family of the quasi-Newton methods, *TRU Mathematics*, vol.16-1, pp.49-54, 1980.

[8] H. Yamashita, A globa11y convergent primal-dual interior point method for constrained optimization, *Optimization Methods and Software*, vol.10, pp.443-469, 1998.

[9] H. Yamashita, H. Yabe and T. Tanabe, A globally and superlinearly convergent primal-dual interior point trust region method for large scale constrained optimization, *Mathematical Programming*, vol.102, pp.111-151, 2005.