一次法としてみた座標降下法，乗数法，交互方向乗数法

山下　信雄

本稿では，大規模な連続最適化問題に対する座標降下法と乗数法，交互方向乗数法を一次法の観点から紹介する．

キーワード：座標降下法，乗数法，交互方向乗数法，L-BFGS 法，部分問題，スケーリング

1. はじめに

近年，データ解析や機械学習などに現れる大規模な連続最適化問題の解法として，一次法が再び注目されている．10 年ほど前までの数理最適化の講義では，一次法は役立たずの手法としてさっと流し，内点法や逐次二次計画法など，いわゆる二次法の解説に重点を置いていた．それが機械学習などの流行によって，一次法の講義の需要が増している．しかし，一次法は問題依存なところがあり，体系だって授業計画を立てることが意外と難しい．本稿では，そのような講義において，最も組み入れにくい（？）座標降下法と乗数法，交互方向乗数法 (ADMM) を，私見を交えながら解説したい．なお，より詳しく知りたい方には文献 [1] をお勧めする．

座標降下法や ADMM を講義に取り入れにくいのは，それらの手法が単純には一次法とみなしにくいからである．そこで，まずは，本稿における一次法の定義から話を進めたい．

図 1 にあるように，数理最適化の手法を利用する際には，まず，数理最適化モデルを構築する．その後，そのモデルの解法を考える．解法として一次法を考えるときの重要なキーワードは**部分問題**と**スケーリング**である．連続最適化の解法の多くは，最適解に収束する点列を生成する反復法である．その反復法では，各反復において，解きたい最適化モデルを近似した問題（以下，部分問題と呼ぶ）を構成し，その最適解を用いて次の反復点を生成する．

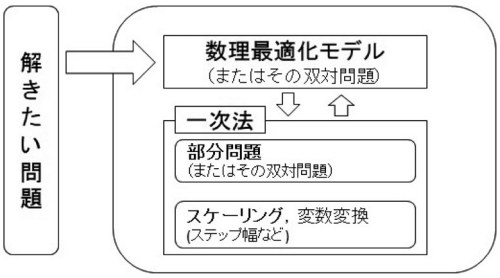

図 1　数理最適化モデルと一次法

$$\begin{array}{lll} \text{最適化モデル} & & \text{部分問題} \\ \min \quad f(x) & \approx & \min \quad \tilde{f}^k(x) \\ \text{s.t.} \quad g(x) \leq 0 & & \text{s.t.} \quad \tilde{g}^k(x) \leq 0 \end{array}$$

部分問題が元の問題によく近似できていれば（$\tilde{f}^k \approx f$, $\tilde{g}^k \approx g$ であれば），その解は元の問題のよい近似解になる．その近似解を用いてより近似されている部分問題を構成すれば，さらによい近似解を得ることができる．そのような部分問題を用いた反復法は少ない反復回数で最適解を見つけられる．一方，その部分問題が元の問題をよく近似できているほど，元の問題と解く手間が変わらなくなる．つまり，反復回数と部分問題の難易度はトレードオフの関係になっている．二次法は部分問題の近似に重点を置いた手法であり，一次法は部分問題の解きやすさに重点を置いた手法といえる．

一般に，部分問題の近似精度によって，一次法，二次法などと分類が行われる．たとえば，f を二次近似した関数 $\tilde{f}^k(x) = f(x) + \mathrm{O}(\|x - x^k\|^2)$ で部分問題を構成したニュートン法は二次法である．一方で，一次法は，最急降下法のように一次近似をそのまま使う手法だけではない．一次よりも劣る近似を使う手法，たとえば劣勾配法や確率勾配法も一次法と呼ぶことがある．そこで，本稿では，二次よりも劣る精度の部分問題を

やました　のぶお
京都大学大学院情報学研究科
〒 606–8501 京都府京都市左京区吉田本町
nobuo@i.kyoto-u.ac.jp

表 1　p 次法

0 次法	一次法	二次法
劣勾配法 確率勾配法	最急降下法 共役勾配法 L-BFGS 法 座標降下法	ニュートン法 準ニュートン法
	射影勾配法 交互方向乗数法	内点法 逐次二次計画法

上段は制約なし，下段は制約付き問題の解法

用いた手法を一次法と呼ぶことにする．なお，生成された点列の最適解への収束率から分類することもできる．目的関数のヘッセ行列などの二次の微分情報を利用しない手法は，高々一次収束しかしない[1]．その高々一次収束しかしない手法を一次法と定義してもよい．

表 1 には代表的な一次法と二次法を掲載した．なお，表中にある 0 次法は，一次の微分情報すら用いない手法（一次収束すらしない手法）であるが，本稿ではこれらも一次法としている．表 1 にある二次法の部分問題は線形方程式や凸二次計画問題となる．それらの部分問題を解くためには，少なくとも $\mathrm{O}(n^3)$ の計算時間が必要である．ただし，n は決定変数の数である．数万変数程度の問題に対しては，それらの部分問題は問題なく解ける．しかし，データ解析などに現れる大規模な問題に対しては，$\mathrm{O}(n^3)$ の計算は許しにくく，1 回の反復もできないことがある．そのような問題に対して一次法の出番となる．制約なしの問題に対しては，最急降下法（近接勾配法）や共役勾配法，BFGS 更新を用いた記憶制限準ニュートン法（以下，L-BFGS 法）が使われる．一方，制約つき問題に対しては，実行可能領域への射影が簡単に計算できるときは射影勾配法，簡単に計算できないようなときは乗数法が使われる．また，問題に特別な構造をもつときには，座標降下法や ADMM が適用できる．

一次法はそこそこの精度の近似解を現実的な時間で求めることができる．しかし，求める精度が高くなると，かなりの反復回数が必要となることがある．部分問題が速く解ける一次法の主な計算負荷は目的関数やその勾配の評価である．一次法においてこの評価回数を減らすために大切となるのは，部分問題の構成とともに，関数や変数の**スケーリング**である．

ここで，目的関数が f の制約なし最小化問題を考えてみよう．このとき，最急降下法は $x^{k+1} = x^k + t_k \nabla f(x^k)$ として[2]，次の反復点 x^k を求める．ここで，t_k はステップ幅である．一方，目的関数を 100 倍した問題では，最適解は同じであるが，生成される点列は $x^{k+1} = x^k + 100 t_k \nabla f(x^k)$ となり，ステップ幅が同じ場合は，異なる点を生成する（二次法であるニュートン法では同じ点が生成される）．このように一次法では，問題に現れる関数や，変数のスケールの違いを十分に考慮していないため，何らかのスケーリングが必要となる．

スケーリング手法の一つがステップ幅 t_k を定める直線探索である．アルミホのルールやウルフのルールなど代表的な直線探索の手法では，1 回の反復中に何度も関数評価をすることがあり，機械学習などの大規模な問題ではなるべく使いたくない．そのため，機械学習などでの一次法では，定数ステップ幅や $t_k \downarrow 0$ とする減衰ルール[3]などが使われる．座標降下法や ADMM は簡単に計算できる定数ステップ幅を用いた手法である．

本稿では，説明を簡単にするために，次の連続最適化問題を考える．

$$\begin{array}{ll} \min & f(x) := h(x) + \sum_{i=1}^{\ell} g_i(x_i) \\ \mathrm{s.t} & Bx = b \\ & x_i \in S_i \ (i = 1, \ldots, \ell) \end{array} \tag{1}$$

ただし，決定変数は $x = (x_1^\top, x_2^\top, \ldots, x_\ell^\top)^\top \in R^n$ であり，$B \in R^{m \times n}, b \in R^m$ である．また，$n_i (i = 1, \ldots, \ell)$ を $n = \sum_{i=1}^{\ell} n_i$ を満たす自然数とし，$S_i \subseteq R^{n_i}$ とする[4]．h は微分可能な関数であり，g_i は凸関数とする．

座標降下法は，各反復において，ある変数 $x_i \in R^{n_i}$ 以外の変数を固定した x_i のみの部分問題を解き，その最適解を次の反復点とする手法である．部分問題の決定変数は x_i だけであるため，高速に解くことができる．乗数法は，制約条件を組み込んだ拡張ラグランジュ関数の制約なし最小化問題を部分問題とする解法である．ADMM は，この部分問題をさらに座標降下法によって解くというアルゴリズムである．以下では，座標降下法，乗数法，ADMM の順番で解説する．その際には，部分問題の構成方法とスケーリングについて注意してほしい．

なお，本稿では，変数 $x_i \in R^{n_i}$ に関する勾配を

[1] x^* を最適解とする．$\|x^k - x^*\| \leq c^k \rho$ となる $\rho > 0$ と $c \in (0, 1)$ が存在するとき $\{x^k\}$ は x^* に一次収束するという．

[2] $\nabla f(x)$ は f の x における勾配を表す．

[3] たとえば $t_k = 1/\sqrt{k}$.

[4] $x_i \in S_i$ としては $x_i \geq 0$ など簡単な制約条件を想定している．

$\nabla_{x_i} f(x)$ と表す．$n_i = 1$ のときは，$\nabla_{x_i} f(x) = \frac{\partial f(x)}{\partial x_i}$ である．

2. 座標降下法

座標降下法は，各反復で座標（変数）を選び，その座標上でのみの最適化問題を部分問題とする手法である [2]．各反復で，座標（変数）を複数選ぶ座標降下法をブロック座標降下法と呼ぶ．以下ではブロック座標降下法も座標降下法と呼ぶ．また，複数選んだ変数の集合をブロックと呼ぶことにする．問題 (1) に対してブロックの候補を $x_i \in R^{n_i} (i = 1, \ldots, \ell)$ とした座標降下法は以下のように表せる．ただし，$B_i \in R^{m \times n_i}$ は $B = (B_1, \ldots, B_\ell)$ とする B の分割である．

座標降下法： $i \in \{1, \ldots, \ell\}$ を選ぶ．

$$\begin{aligned} \min \quad & f(x_1^k, \ldots, x_i, \ldots, x_\ell^k) \\ \text{s.t.} \quad & x_i \in S_i \\ & B_1 x_1^k + \cdots + B_i x_i + \cdots + B_\ell x_\ell^k = b \end{aligned} \tag{2}$$

の最適解（あるいは近似解）を x_i^{k+1} とし，$x_j^{k+1} = x_j^k$ $(j \neq i)$ とする．

ブロック数が 2 となる座標降下法を特に交互方向法と呼ぶことがある．なお，ここでは簡単のため，ブロックの候補は予め与えているが，反復の途中にブロックの構成を変えても構わない．

目的関数 f が微分可能な関数で，等式制約がない場合は，生成された点列は停留点に収束する．特に，目的関数が全変数に関して強凸関数[5]でなくても，各ブロックに関して強凸関数であれば一次収束することが示されている [3]．一方で，f が微分可能でなかったり，各 x_i が共通の等式制約をもつときには収束しないことがある．たとえば次の問題を考えてみよう．

$$\begin{aligned} \min \quad & x_1 \\ \text{s.t.} \quad & x_1 + x_2 = 1 \\ & x_1 \geq 0,\ x_2 \geq 0 \end{aligned} \qquad \begin{aligned} \min \quad & x_1^2 + x_2^2 + 3|x_1 + x_2| \end{aligned}$$

左の問題の最適解は $(0, 1)^\top$ であり，右の問題の最適解は $(0, 0)^\top$ である．左の問題は等式制約を満たすため 1 変数だけでは更新できない．右の問題では $x^0 = (1, -1)^\top$ としたとき更新が行われない．これは $|x_1 + x_2|$ が微分可能でないためである．また，更新ができる場合でも，循環して収束しない例も知られている [2]．等式制約がある場合は，等式制約の数よりも多い変数のブロックを選ばなければならない．また，微分不可能な関数を含むときは，$\sum_{i=1}^{\ell} g_i(x_i)$ というようにブロックごとに分解可能な構造をしていなければならない．

2.1 部分問題

部分問題 (2) は，Bregman 関数を用いて次のように一般化できる．

$$\begin{aligned} \min \quad & \nabla_{x_i} h(x^k)^\top (x_i) + g_i(x_i) + D_{\psi_k}(x_i, x_i^k) \\ \text{s.t.} \quad & x_i \in S_i \\ & B_1 x_1^k + \cdots + B_i x_i + \cdots + B_\ell x_\ell^k = b \end{aligned} \tag{3}$$

ここで，D_ψ は微分可能な凸関数 ψ を用いて $D_\psi(u, v) = \psi(u) - \psi(v) - \nabla\psi(v)^\top (u - v)$ と定義される Bregman 関数であり，$\psi(u) = \frac{1}{2}\|u\|^2$ のときは，$D_\psi(u, v) = \frac{1}{2}\|u - v\|^2$ となるため，距離（の 2 乗）の一般化とみなすことができる．なお，Bregman 関数を用いた一般化は，最急降下法に対する鏡像降下法への一般化と同じであり，一次法ではよく用いられる．また，ψ_k は部分問題のスケーリングと考えることができる．ここで $\psi_k(x_i) = f(x_1^k, \ldots, x_i, \ldots, x_\ell^k)$ とおくと，部分問題 (3) は部分問題 (2) となる．理論的な収束性は，$\nabla_{x_i} f$ のリプシッツ定数を L_i としたとき，$B_{\psi_k}(x_i, x_i^k) \geq \frac{L_i}{2}\|x_i - x_i\|^2$ であれば，部分問題 (2) を用いた場合と変わらない．

2.2 ブロック $x_i, i \in \{1, \ldots, \ell\}$ の選び方

ブロックの選び方には次の三つの方法がある．

（ほぼ）巡回ルール：$r \geq \ell$ とする．r 回反復した際には，各 x_i を必ず 1 回は選ぶルール．$1, 2, 3, \ldots, \ell$ と順番に選ぶ古典的な巡回ルールを含んでいる．また，ℓ 回ごとに添え字をシャッフルして巡回ルールを行う**シャッフルルール**もこのルールに含まれる．

乱択ルール：ブロック x_i をランダムに選ぶルール．x_i の選ばれる確率が均一なものと，目的関数の性質に応じて確率を変えるものがある．

貪欲ルール：最も目的関数が減少すると予想される x_i を選ぶルール．各ブロックに関する部分問題 (2) をすべて正確に計算して最善のものを選ぶ方法と，近似的に行う方法がある．近似的に行う方法としては，最適性の条件（制約なしの場合 $\nabla f(x) = 0$）から，その条件を最も満たしていない（$\|\nabla_{x_i} f(x)\|$ が最も大きい）ブロックを選ぶ方法がある．

どのルールでも，大域的収束[6]することが示されている [2]．ただし，乱択ルールによるものは，期待値を用

[5] 任意の x, y と $\alpha \in [0, 1]$ に対して $f(\alpha x + (1 - \alpha)y) + \mu\alpha(1 - \alpha)\|x - y\|^2 \leq \alpha f(x) + (1 - \alpha) f(x)$ が成り立つとき f を強凸関数という．

[6] どのような x^0 から始めても停留点（制約なしの場合は $\nabla f(x^*) = 0$ となる x^*）に収束すること．

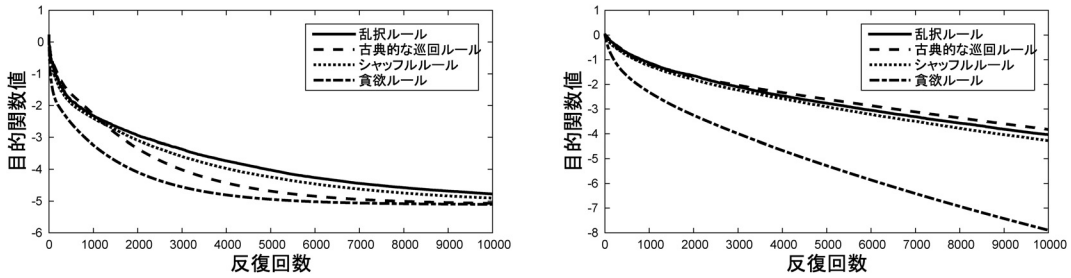

図 2　ブロック選択
左 $n = 10$，右 $n = 100$

いて解析されている．乱択ルールの収束性は，理論的にも実験的にも，古典的な巡回ルールよりもよい．一方で，シャッフルルールは，実験的には乱択ルールよりもよいことが報告されている．しかし，その理由は理論的にはまだ解明されていない．反復回数で評価する場合は，貪欲ルールが一番優れたルールである．しかし，最善なブロックを選ぶのに時間がかかることがある．

図 2 は，上記の四つのルールを用いて，簡単な凸二次関数の最小化を行った結果である．ただし，$n_i = 1$ としている．左図は $n = 10$ のとき，右図は $n = 100$ のときである．変数が多いとき，貪欲ルール，シャッフルルール，乱択ルールの順で結果がよい．一方で，変数が少ないとき，古典的な巡回ルールがよい結果を出している．これは，乱択ルールでは，同じブロックが続けて選ばれたり，逆にしばらく選ばれないブロックがあるためだと思われる．なお，この座標降下法では，n 回の反復の計算量が，最急降下法や共役勾配法の 1 回の反復の計算量と同じになる．凸二次関数の最小化では共役勾配法が高々 n 回の反復で最適解が求まることに注意すると，座標降下法では n^2 の反復で最適解を求めてほしい（$n = 10$ のときは 100 回目の反復，$n = 100$ のときは 1 万回目の反復）．図から見てもわかるように，座標降下法は極めて遅いことがわかる．これは座標降下法では，変数全体でのスケーリングができていないためである．

なお，等式制約があるときには，各反復でブロックの構成を変える貪欲ルールを使うことが一般的である．これは，ブロックの変数の数 n_i は等式の数 m よりも多く選ばなければならないため，ブロックの候補数 ${}_nC_{n_i}$ は n よりも大きくなり，それらの候補に対して，巡回ルールや乱択ルールを適用することが難しいからである．一方，貪欲ルールでは大域的収束することが証明されている [4]．ただし，$m \geq 3$ のときに貪欲ルールでブロックを構成するには $\mathrm{O}(n^2)$ 以上の計算時間が必要となり，大規模な問題では現実的ではない [4]．

2.3　実装方法

座標降下法の各反復の計算量は，全変数を更新する一次法の計算量をブロックの候補数 ℓ で割った量になるのが理想である．たとえば，最急降下法で 1 反復の計算時間が勾配の計算も含めて $\mathrm{O}(n^2)$ となるときには，1 変数のみ更新する座標降下法では $\mathrm{O}(n)$ となってほしい．以下では簡単のため，凸関数 $\theta : R \to R$ を用いて $f(x) = \sum_{t=1}^{T} \theta(a_t^\top x)$ と表されており，制約条件がない場合の部分問題の解き方を考えよう．ただし，$a_t \in R^n$ であり，機械学習などでは t 番目のデータを表す．このとき，一変数 x_i の最適化は $\nabla_{x_i} f(x_i) = \sum_{t=1}^{T} (a_t)_i \theta'(a_t^\top x) = 0$ を満たす x_i を求めることになる．x_i 以外の変数が x_j^k に固定されているとき，$a_t^\top x = (a_t)_i (x_i - x_i^k) + a_t^\top x^k$ となり，$a_t^\top x^k$ $(t = 1, \ldots, T)$ が計算されていれば，$\mathrm{O}(T)$ で目的関数や勾配を計算できる．この一変数の問題をニュートン法で解けば，その反復回数は数回程度であるから，部分問題を解く計算量は $\mathrm{O}(T)$ とみなせる．次に，$a_t^\top x^k$ の計算を考えよう．この計算を反復ごとにしていたら，$\mathrm{O}(Tn)$ かかってしまい，所望の $\mathrm{O}(n)$ とはならない．しかし，$\eta_t^k := a_t^\top x^k$ を記憶しておけば，$\eta_t^{k+1} = (a_t)_i (x_i^{k+1} - x_i^k) + \eta_t^k$ より $\mathrm{O}(1)$ で計算できる．よって，$a_t^\top x^k$ $(t = 1, \ldots, T)$ の計算量も $\mathrm{O}(T)$ となる．以上より，$T \approx n$ であれば，部分問題は $\mathrm{O}(n)$ で解ける．

2.4　うまくいっている座標降下法の例

上記のように座標降下法は収束が遅く，実装が難しい．そのため，問題に特殊な構造がない場合は，あまりお勧めできない．一方で，以下のように，問題の構造を利用することによって，うまくいく例もある．

混合ガウス分布推定のための EM アルゴリズム：あるデータが混合ガウス分布 $\sum_{i=1}^{n} \alpha_i \mathcal{N}(x_t \mid \mu_i, V_i)$ に従うと予想されるとき，与えられたデータから混

合ガウス分布の各種パラメータを推定したい．ここで，α_i は $\sum \alpha_i = 1$ となる非負のパラメータであり，$\mathcal{N}(x \mid \mu, V)$ は平均 μ，分散共分散行列 V の正規分布の確率密度関数である．これらのパラメータの最尤推定（最適化問題）によく用いられる手法が EM アルゴリズムである．この対数尤度最大化問題は次の非凸な関数の最小化問題とみることができ，EM アルゴリズムはそれに対する座標降下法とみなすことができる [5].

$$\sum_{i=1}^{m}\sum_{t=1}^{T} W_{it}\left(\ln W_{it} - \log \alpha_i - \mathcal{N}(x_t \mid \mu_i, V_i)\right)$$

ここで，決定変数は α_i と μ_i, V_i, W_{it} である．EM アルゴリズムは (α_i, μ_i, V_i) と W_{ik} をブロックと考えた座標降下法である．座標降下法としてみなすことによって，座標降下法の収束性の結果が EM アルゴリズムに適用できる．また，スパース性やパラメータの上下限制約をいれるなど，モデルを拡張した場合でも，座標降下法としてみれば，簡単に EM アルゴルズムを拡張することができる [6].

サポートベクターマシンの SVMLight：サポートベクターマシン (SVM) は機械学習における分類モデルの一つであり，そこでは，以下のような大規模な凸二次計画問題を解く．

$$\begin{array}{ll}\min & \frac{1}{2}x^\top K x \\ \text{s.t.} & \sum_{i=1}^{n} y_i x_i = 0,\ 0 \le x_i \le C\ (i = 1, \ldots, n)\end{array}$$

この SVM では，K は半正定値対称行列，C は正の定数，y_i は -1 か 1 の定数である．等式制約を一つもつため，$n_i = 1$ とした座標降下法を適用できない．最小のブロックである $n_i = 2$ とした座標降下法の実装コードが SVMLight である．$n_i = 2$ としたとき，部分問題は 2 変数の最小化問題となるが，等式制約から 1 変数に減らすことができるため，区間上での二次関数の最小化となり，直ちに最適解を求めることができる．SVMLight では貪欲ルールによってブロックを選んでいる．SVM の最適解ではほとんどの変数が 0 となる．この貪欲ルールでは，0 となる変数は選ばれないので，比較的少ない反復回数で良好の近似解を求めることができる．

線形計画問題の単体法：線形計画の単体法では，各反復において，基底変数とピボット選択された非基底変数をブロックとした小規模の線形計画問題を解いている．つまり，座標降下法の一種である．そこでのブロックの選び方は，相対コスト係数を上手に使った貪欲ルールとみなすことができる．

3. 乗数法

乗数法は制約つき最適化問題の解法であり，その部分問題は拡張ラグランジュ関数の制約なし（あるいは簡単な制約条件をもつ）最適化問題となる．部分問題は非線形な問題となるため，凸二次計画問題などよりも難しいと思われるかもしれないが，制約条件が簡単なため射影勾配法などの一次法を適用することができる．大規模で複雑な制約条件をもっていても，部分問題を近似的に解くことを許せば，現実的な時間でそこそこの近似解を得ることができる．ただし，非凸な問題や複雑な不等式制約を含む問題に対しては，効率的な解法として実装することは難しいようである．以下では，簡単のため凸となる場合の問題 (1) を用いて解説する．

問題 (1) のラグランジュ関数と拡張ラグランジュ関数を以下のように定義する．

$$L(x, \lambda) = f(x) + \lambda^\top (Bx - b)$$
$$L_\beta(x, \lambda) = L(x, \lambda) + \frac{\beta}{2}\|Bx - b\|^2$$

問題 (1) の双対問題は以下のように与えられる．

$$(D) \quad \max_{\lambda \in R^m} \quad \inf_{x \in S} L(x, \lambda) \tag{4}$$

ただし，$S = S_i \times \cdots \times S_\ell$ である．一方，拡張ラグランジュ関数を用いた次の問題も双対問題となる [7].

$$(\mathrm{D}_{L_\beta}) \quad \max_{\lambda \in R^m} \quad \inf_{x \in S} L_\beta(x, \lambda) \tag{5}$$

実際，(D_{L_β}) の最大値は，問題 (1) の最小値以下であり，なおかつ，双対問題 (D) の最大値以上となる．このことから，(1) と (D) の間に双対性ギャップがなければ，(1) と (D_{L_β}) の間にもないことがわかる．

いま，(D_{L_β}) の目的関数を $\omega_\beta(\lambda) := \inf_{x \in S} L_\beta(x, \lambda)$ と定義する．$\inf_{x \in S} L_\beta(x, \lambda)$ の解が唯一解 $x(\lambda)$ をもてば，ω_β は微分可能で，

$$\nabla \omega_\beta(\lambda) = Bx(\lambda) - b$$

となる [7]．双対問題 (5) に対する最急降下法は，ステップ幅を $\tau\beta$ とすると，

$$\begin{aligned} x^{k+1} &= \mathrm{argmin}_{x \in S} L_\beta(x, \lambda^k) \\ \lambda^{k+1} &= \lambda^k + \tau\beta(Bx^{k+1} - b) \end{aligned} \tag{6}$$

と書ける．つまり，乗数法は双対問題 (5) に対する一次法とみなせる．この乗数法は $\tau \in (0, 2)$ であれば大域的収束する．

一方，乗数法はある双対問題に対する近接点法とみなすこともできる [1]．そのため，パラメータ β を反復ごとに大きくしていくと，近接点法の性質から，最適解へ超一次収束[7]することを導くことができる．（この場合は一次法とはいえない．）

乗数法における計算のネックは，x^{k+1} の計算である．座標降下法のように Bregman 関数を用いて一般化することもできる．

$$x^{k+1}=\mathrm{argmin}_{x\in S}\nabla_x L_\beta(x,\lambda^k)^\top(x-x^k)+D_\psi(x,x^k)$$

この場合でも，ψ を適切に選べば大域的収束する．この ψ が部分問題の難しさやスケーリングに関係している．たとえば $\psi(u)=\frac{L}{2}\|u\|^2$ とすれば部分問題は簡単に解くことができ，L が十分大きければ大域的収束する．しかし反復回数は部分問題 (6) を解く場合に比べて大幅に増えてしまう．

4. 交互方向乗数法

交互方向乗数法（以下 ADMM）は，乗数法における主変数の部分問題 (6) を座標降下法で解く手法である．より詳しい解説やその応用問題については [8] や本特集のほかの記事を読んでほしい．以下では，簡単のため，制約条件がない問題 (1) を考える．この問題はスラック変数 $y\in R^n$ を用いて

$$\begin{array}{ll}\min & h(x)+g(y)\\ \mathrm{s.t} & x-y=0\end{array} \tag{7}$$

と表せ，その拡張ラグランジュ関数は

$$L_\beta(x,y,\lambda)=h(x)+g(y)+\lambda^\top(x-y)+\frac{\beta}{2}\|x-y\|^2$$

となる．乗数法では，

$$(x^{k+1},y^{k+1})\in\mathrm{argmin}_{x,y}L_\beta(x,y,\lambda^k)$$

として，(x^{k+1},y^{k+1}) を求めるが，拡張ラグランジュ関数の最後の項 $\frac{\beta}{2}\|x-y\|^2$ のせいで，x と y を分けて解くことができない．一方，応用によっては，x あるいは y を固定した場合，固定されていない変数の問題は簡単に解けることがある．そこで，部分問題を正確に解く代わりに，座標降下法（交互方向法）を 1 回実行するのが ADMM である．

交互方向乗数法 (ADMM)

$$x^{k+1}=\mathrm{argmin}_x L_\beta(x,y^k,\lambda^k) \tag{8}$$
$$y^{k+1}=\mathrm{argmin}_y L_\beta(x^{k+1},y,\lambda^k) \tag{9}$$
$$\lambda^{k+1}=\lambda^k+\tau\beta(Ax^{k+1}-y^{k+1})$$

ADMM は $\tau\in(0,(1+\sqrt{5})/2)$ のとき大域的に収束する．ADMM は，ブロックの数が三つ以上の問題に対しても，拡張が可能なように思われるかもしれない．しかしながら，ブロック数が 3 のときに $\tau=1$ とした場合，収束しない例が報告されている [9]．大域的収束のためには，目的関数に何かしらの仮定を置くか，アルゴリズムの変更が必要になる．たとえば，τ をブロック数に応じて十分小さく選べば，大域的収束することが知られている [10]．ただし，収束は非常に遅くなる．

4.1 部分問題

部分問題 (8) と (9) の解き方を考えてみよう．以下では簡単のため，$h(x)=\frac{1}{2}\|Ax-a\|^2$，$g_i(y_i)=|y_i|$ とした最適化問題を考える．ただし，$A\in R^{T\times n}$，$b\in R^T$ である．このとき，y の部分問題 (9) は $\mathrm{O}(n)$ で解ける．一方，x の部分問題 (8) の解は

$$x^{k+1}=(A^\top A+\beta I)^{-1}(-\lambda^k+\beta y^k+A^\top a) \tag{10}$$

となる．ここで，$n\times n$ 行列 $(A^\top A+\beta I)^{-1}$ は各反復で変わらないことに注意すると，一度，コレスキー分解をしておけば，各反復の計算は高速にできる．$T\ll n$ のときは，Sherman–Morrison の公式：

$$(A^\top A+\beta I)^{-1}=\frac{1}{\beta}I-\frac{1}{\beta^2}A^\top\left(I+\frac{1}{\beta}AA^\top\right)^{-1}A$$

より，小規模な $T\times T$ 行列 $(I+(1/\beta)AA^\top)$ のコレスキー分解を用いても x^{k+1} を計算できる．

大規模な問題においては，1 回のコレスキー分解も大変な場合がある．そのような場合，Bregman 関数を用いて一般化した近接 ADMM を考えてもよい．以下では，x の部分問題だけを考える．y の部分問題に対しても同様に一般化できる．

$$x^{k+1}=\mathrm{argmin}_x\left\{L_\beta(x,y^k,\lambda^k)+D_\phi(x,x^k)\right\}$$

ここで，$\phi(u)=\frac{1}{2}u^\top Pu$ とし，P が半正定値対称行列の場合を考えよう．このとき，$D_\phi(x,x^k)=\frac{1}{2}(x-x^k)^\top P(x-x^k)$ となり，x^{k+1} は次式で与えられる．

$$x^{k+1}=x^k-(P+\beta I+A^\top A)^{-1}\nabla_x L_\beta(x^k,y^k,\lambda^k). \tag{11}$$

ここで，$P:=\gamma I-\beta I-A^\top A$，$\gamma>0$ とすれば，

[7] x^* を最適解としたとき，$\|x^{k+1}-x^*\|\le c_k\|x^k-x^*\|$ かつ $c_k\to 0$ となる非負の数列 $\{c_k\}$ が存在するとき，$\{x^k\}$ は x^* に超一次収束するという．

$(P+\beta I+A^\top A)^{-1}=\frac{1}{\gamma}I$ となり，部分問題は簡単に解ける．また，γ を行列 $\beta I+A^\top A$ の最大固有値よりも大きくとれば，P は半正定値行列となる．このとき，この ADMM は大域的収束する．なお，この計算は部分問題 (8) に対して，最急降下法（一次法）を一回しているとみなすことができる．ただし，二次の情報（スケーリング）が使われていないため，収束は遅くなることがある．そのような欠点を克服するため，P を BFGS 法や L-BFGS 法を利用して構築することが考えられている [11]．その基本となるアイデアは，γI の代わりに $\beta I+A^\top A$ の近似行列 B_k を用いることである．

そのような ADMM を紹介する前に，まず，目的関数 f の制約なし最小化問題の解法としての BFGS 法と L-BFGS 法を簡単に説明する．BFGS 法は，準ニュートン法の一つで，ヘッセ行列 $\nabla^2 f(x^k)$ の近似行列 B_k の逆行列 H_k を用いて，$x^{k+1}=x^k-t_kH_k\nabla f(x^k)$ として点列を生成する．行列 H_k は，行列 $V_k=I-\frac{s_ky_k^\top}{s_k^\top y_k}$, $S_k=\frac{s_ks_k^\top}{s_k^\top y_k}$ を用いて，$H_{k+1}=V_kH_k(V_k)^\top+S_k$ と更新する．ただし，$s_k=x^{k+1}-x^k$, $y_k=\nabla f(x^{k+1})-\nabla f(x^k)$ である．一般に，H_k は密な行列となるため，BFGS 法は大規模な問題には適用できない．そのような欠点を克服するために提案された手法が L-BFGS 法である．BFGS 法による H_{k+1} は

$$\begin{aligned}H_{k+1}=&V_kH_kV_k^\top+S_k\\=&V_kV_{k-1}H_{k-1}V_{k-1}^\top V_k^\top+S_k+V_kS_{k-1}V_k^\top\\&\vdots\end{aligned}$$

と展開できるため，任意の $0\le r\le k$ となる整数 r に対して，H_{k-r} と $r+1$ 個のペア $(s_{k-r},y_{k-r}),\dots,(s_k,y_k)$ を用いて，H_{k+1} に関する演算を行うことができる．L-BFGS 法では，H_{k-r} の代わりに対角行列，たとえば $(\|s_k\|^2/y_k^\top s_k)I$ を用いることによって，H_{k+1} を構成する．その結果，L-BFGS 法が必要とする記憶容量は $O(rn)$ ですむ．さらに，探索方向 $-H_k\nabla f(x^k)$ の計算も，V_k,S_k がベクトルの積で表現されていることを利用して，$O(rn)$ でできる．L-BFGS 法は，H_{k-r} を対角行列で近似しているため，高々一次収束しかしない一次法である．問題構造を特に利用できないような制約なし最適化問題に対して，非線形共役勾配法とともによく用いられる手法である．

話を ADMM に戻そう．ADMM において，$P=B_k-\beta I-A^\top A$ とすることを考える．このとき，更新式 (11) において $(P+\beta I+A^\top A)^{-1}=B_k^{-1}=H_k$ となり，H_k が計算されていれば，x^{k+1} を容易に求めることができる．P は，通常の ADMM の部分問題 (10) に近づけるためには $P\approx 0$ であることが望ましく，大域的収束のためには半正定値行列となることが望ましい．そこで，B_k を $M:=\beta I+A^\top A$ の近似行列となるように計算する．つまり，$s=x^{k+1}-x^k$, $y=Ms$ として，BFGS 法によって B_k を構築する．また，P の半正定値性に関連して，次の定理が示されている [11]．

定理 1. $s\neq 0$ とし，$y=Ms$ とする．このとき，$M\preceq B_k$ であれば $M\preceq B_{k+1}$ である．

この定理より，B_0 を $M\preceq B_0$ と選べば，すべての k に対して $P=B_k-M$ は半正定値行列となる．この P を用いた ADMM の数値実験では，反復回数は通常の ADMM の部分問題 (10) よりは若干増えるが，最急降下法によるものの半分ぐらいになることが報告されている [11]．

ところで，ADMM が大域的収束性をもつためには，ブロックを二つにとることが重要であった．元の数理最適化モデルが二つのブロックで表されていないときも，適当な変数を導入することで，二つのブロックの部分問題を構成できる場合がある．そのときのキーワードが 2 部グラフである．まずは，次の問題を考えてみよう．

$$\min\ \sum_{t=1}^T f_t(A_tx)$$

この問題は，各関数 f_t に共通の変数 x を含んでいる．A_t をデータだと考えれば，信号処理などによく現れる最適化問題である．各 t に変数 y_t を用意することによって，次の等価の問題に変換できる．

$$\begin{array}{ll}\min & \sum_{t=1}^T f_t(y_t)\\ \text{s.t} & y_t=A_tx\ (t=1,\dots,T)\end{array}\qquad(12)$$

一見，$x,y_1,\dots,y_T$ と $T+1$ 個のブロックがあるように見えるが，$y=(y_1,\dots,y_T)$ とすれば，x と y の二つのブロックの問題となる．このとき，拡張ラグランジュ関数は，

$$\begin{aligned}&L_\beta(x,y_1,\dots,y_T,\lambda_1,\dots,\lambda_T)\\=&\sum_{t=1}^T f_i(y_t)+\sum_{t=1}^T\lambda_t^\top(A_tx-y_t)+\frac{\beta}{2}\sum_{t=1}^T\|y_t-A_tx\|^2\end{aligned}$$

となり，ADMM における y の部分問題は，以下のように t ごとに分解できる．

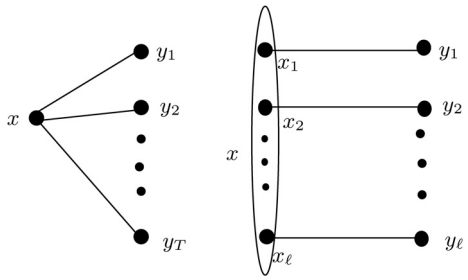

図 3　部分問題の変数の関係
左は (12)，右は (13)

$$y_t^{k+1}=\min_{y_t} f_t(y_t)+\lambda_t^\top(A_t x^k-y_t)+\frac{\beta}{2}\|y_t-A_t x^k\|^2$$

そのため，ブロック y_t ごとに解くことと，それらを一つにまとめて y として解くことが同一視できる．

次に，複数のブロックで共通の等式制約をもつ次の問題をみてみよう．

$$\begin{array}{ll}\min & \sum_{i=1}^{\ell} f_i(x_i)\\ \text{s.t} & \sum_{i=1}^{\ell} B_i x_i = b\end{array}$$

これは，輸送問題などによく現れる問題である．この問題も，$y_i(i=1,\ldots,\ell)$ を用意することによって，次の等価な問題に変換できる．

$$\begin{array}{ll}\min & \sum_{i=1}^{\ell} f_i(y_i)\\ \text{s.t} & \sum_{i=1}^{\ell} B_i x_i = b\\ & y_i = x_i\ (i=1,\ldots,\ell)\end{array}\qquad(13)$$

この問題においては，$x=(x_1,\ldots,x_\ell)$ と $y=(y_1,\ldots,x_\ell)$ の二つのブロックを考えればよい．このときも，y に対しては y_i ごとに独立した部分問題に分解することができる．ただし，x は等式制約のため，x_i ごとに分解することはできない．そのため，大規模な問題においては，上記の L-BFGS 法を用いた手法などを利用して x の部分問題を解く必要がある．

これらの例からわかることは，部分問題としたときに互いに独立な変数は一つのグループにまとめることができるということである．乗数法の部分問題 (6) の変数の関係を図 3 のようにグラフで表したとき，2 部グラフとして表せれば，ADMM を適用できる．逆に，ADMM を適用できるように，変数の関係が 2 部グラフとなる最適化モデルを再構成することを考えてもよい．

5.　たかが一次法，だから難しい（面白い）…

一次法は高々一次収束しかしない．そのため，大規模な問題に対して，そこそこの近似解を現実的な時間内で得るための手法と考えたほうがよい．また，一次法のユーザの多くは理論的な収束性より，実際の結果に重きを置いている．たとえば，ニューラルネットワークの学習によく使われる確率勾配法やその工夫版である Adam は，凸最適化問題においてリグレットなどの理論的な収束性が解明されているが，ニューラルネットワークは凸な問題ではない．確率勾配法が使われるのは，理論的によい収束性があるからではなく，使える手法だからである．無論，よい学術誌に掲載されるためには，理論的な結果が必要である．一次法の“研究”の難しさは，理論と実践にギャップがあるためだと思われる．

一次法の開発の難しさ（と面白さ）はスケーリングと実装にある．一次法はスケーリングを上手にしないと非常に遅くなる．乗数法や ADMM でいえばパラメータ β と τ の調整である．適切なパラメータは問題を解くまでわからない．ユーザによってはそこまで気が回らず，その一次法を使えない手法とみなしてしまうかもしれない．また，アルゴルズムとしては同一であっても，実装によって計算時間が大幅に変わることがある．たとえば，ADMM の部分問題 (10) は，$T \ll n$ のときに Sherman–Morrison の公式を使うだけで，桁違いに速くなる．また，本稿では扱っていないが，一次法では射影や近接オペレータの効率的な計算も大切である．たとえば，単体 $\{x \mid \sum x_i = 1,\ x \geq 0\}$ への射影は中央値アルゴリズムを使えば $O(n)$ でできるが，そのことはあまり知られていない．

一方，コーディングの仕方によって，計算速度が数十倍，時には数百倍も異なることがある．特にメモリ管理や並列計算が重要となる．CPU のキャッシュメモリを有効に利用するかどうかで，速さに数十倍の差がでることがある．Python や Matlab などの高級言語ではそのような管理ができないため，優れた一次法や計算テクニックを考案したとしても，うまくコーディングされた既存の手法に負けてしまうことがある．たとえば，SVM のソルバーである SVMLight はうまくコーディングされており，適当にコーディングした一次法では太刀打ちできない．薄型テレビの歴史において，液晶よりも優れている方式がいくつか誕生したが，液晶の生産技術の発展に負けてしまい普及しなかったものがある．一次法においても，コーディングの問題によって日の目を見ないアルゴリズムがあるかもしれない．

一次法は，二次法ほど汎用的な手法ではなく，ピタッとハマった問題に対してしか，よい性能を発揮できない．そのため，そもそも，どの一次法を選ぶのか，あ

るいは使う一次法に併せて，どのように数理最適化モデルを再構築するかが大切となる．

このように一次法は，数理最適化の知識だけでなく，モデリング，数値計算，アルゴリズム論，プログラミング技術を総動員しないと，よいものができない．一方で，いったんよいと認められると，多くのユーザに使ってもらえるものとなるため，研究・開発のしがいがある．

最後に「はじめに」に戻るが，一次法は研究としては面白いが，講義で扱うのは難しい．本稿も体系的にまとめようと頑張ったつもりではあるが，扱う問題がコロコロと変わり，わかりづらかったと思う．現在の一次法の需要を考えると授業でもっと時間を割いたほうがよいのであろう．しかし，二次法を削ってまでやることなのか悩ましいところである．すぐそこに，二次法の逆襲がきているかもしれないし．

参考文献

[1] 金森敬文，鈴木大慈，竹内一郎，佐藤一誠，『機械学習のための連続最適化』，講談社，2016.

[2] S. J. Wright, “Coordinate descent algorithms,” *Mathematical Programming*, **151**, pp. 3–34, 2015.

[3] X. Hua and N. Yamashita, “Block coordinate proximal gradient methods with variable Bregman functions for nonsmooth separable optimization,” *Mathematical Programming*, **160**, pp. 1–32, 2016.

[4] P. Tseng and S. Yun, “A coordinate gradient descent method for linearly constrained smooth optimization and support vector machines training,” *Computational Optimization and Applications*, **47**, pp. 179–206, 2010.

[5] R. J. Hathaway, “Another interpretation of the EM algorithm for mixture distributions,” *Statistics & Probability Letters*, **4**, pp. 53–56, 1986.

[6] Y. Yamakawa and N. Yamashita, “A block coordinate descent method for maximum likelihood estimation problems of mixture distributions,” *Pacific Journal of Optimization*, **11**, pp. 669–686, 2015.

[7] 福島雅夫，『非線形最適化の基礎』，朝倉出版，2001.

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*, Foundations and Trends in Machine Learning, Now Publishers, 2011.

[9] C. Chen, B. He, Y. Ye and X. Yuan, “The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent,” *Mathematical Programming*, **155**, pp. 57–79, 2016.

[10] M. Hong and Z.-Q. Luo, “On the linear convergence of the alternating direction method of multipliers,” *Mathematical Programming*, **162**, pp. 165–199, 2017.

[11] Y. Gu and N. Yamashita, “An alternating direction method of multipliers with the BFGS update for structured convex quadratic optimization,” arXiv: 1903.02270, 2019.