機械学習問題における確率的最適化技法

鈴木　大慈，二反田　篤史，村田　智也

本稿では，機械学習における確率的最適化手法を取り上げる．確率的最適化は大規模データを用いた機械学習を高速に実行するために有用であり，特に一次法との相性がよい．本稿では，その中でもわれわれが提案してきた二つの手法「確率的 DC 計画法」および「二重加速確率的分散縮小勾配降下法」を紹介する．

キーワード：確率的最適化，DC 計画，確率的分散縮小勾配降下法，Nesterov の加速法，機械学習

1. はじめに

本稿では，機械学習問題における確率的最適化手法を二つ紹介する．機械学習において確率的最適化は，目的関数に現れる有限和や積分をランダムサンプリングで置き換えながら最適化する方法である．ランダムサンプリングを用いることで，有限和や積分を正確に計算する必要がなく，更新にかかる計算時間を短縮でき，大規模データを用いた学習などを効率的に実行できる．本稿では，まず DC (difference of convex functions) 計画における確率的最適化手法 [1]（2 節）を紹介し，続いて経験誤差最小化にて有用な確率的分散縮小勾配降下法の加速法を紹介する [2]（3 節）．機械学習においては，大規模データを扱ったり，多数の単純な関数の和を最小化することが多い．そのような場面において確率的最適化手法は強力な手法である [3]．その中でも，DC 計画はボルツマンマシンや隠れ変数モデルで現れる重要な問題設定である．ここで紹介する手法を用いることで，既存手法よりも効率的な最適化が可能になる．後半で扱う経験誤差最小化は，高次元スパース学習において重要なスパース正則化学習などで現れる標準的な問題である．紹介する手法を用いることで，最小のミニバッチサイズで最適な反復回数を達成することができる．

すずき　たいじ
東京大学大学院情報理工学系研究科
〒 113–8656 東京都文京区本郷 7–3–1
taiji@mist.i.u-tokyo.ac.jp
理化学研究所革新知能統合研究センター
〒 103–0027 東京都中央区日本橋 1–4–1
にたんだ　あつし
東京大学大学院情報理工学系研究科
〒 113–8656 東京都文京区本郷 7–3–1
nitanda@mist.i.u-tokyo.ac.jp
理化学研究所革新知能統合研究センター
〒 103–0027 東京都中央区日本橋 1–4–1
むらた　ともや
株式会社 NTT データ数理システムシミュレーション＆マイニング部
〒 160–0016 新宿区信濃町 35 信濃町煉瓦館 1 階
murata@msi.co.jp

2. 確率的 DC 計画法

本節では，DC 計画に対する確率的最適化手法を紹介する．ここで紹介する手法は，文献 [1] で提案されたものである．

2.1 問題設定

DC 計画 [4] は次の形で定式化される：

$$\underset{x\in\mathbb{R}^d}{\text{minimize}}\ \ f(x) \overset{\text{def}}{=} g(x) - h(x), \tag{1}$$

ただし，g と h は $\mathbb{R}^d \to \mathbb{R}$ なる微分可能な凸関数である．

DC 構造はさまざまな機械学習応用において現れる重要な構造である．たとえば，経済，金融，オペレーションズ・リサーチ，生物学といった応用で用いられている．より機械学習的な問題においても，マルチプルカーネル学習 [5] やサポートベクトルマシンにおける特徴選択問題 [6] において DC 計画が現れる．さらには，ボルツマンマシン (Boltzmann machine, BM) といった重要な応用も存在する．これは，二値変数を観測値と隠れ変数にもち，エネルギー関数を用いて定式化された生成モデルである．さらに，DC 構造は次のような性質をもつ：(i) Stone–Weierstrass の定理と多項式の DC 分解により，コンパクト集合上の任意の連続関数は DC 関数により近似可能である [7–9]；(ii) ヘッセ行列が下に有界な任意の C^2-関数は DC 関数として表現できる．

最適化問題 (1) を解くための代表的な手法として DC アルゴリズム (DC algorithm, DCA) とその変種がある [4]．これは，部分問題として凸関数 g と凹関数 $-h$ の線形近似の和を最小化する問題を考え，各反復にこの部分問題を解くというものである．DCA はその定

表 1　SPD の計算量

	一般的設定	滑らかな h	Polyak–Łojasiewicz 条件
外側反復数	$\mathrm{O}(L_g/\epsilon)$	$\mathrm{O}(\min\{L_g, L_h\}/\epsilon)$	$\mathrm{O}(CL_g \log \frac{1}{\epsilon})$
総計算量（一般論）	$\mathrm{O}(L_g/\epsilon^2)$	$\mathrm{O}(L_g/\epsilon^2)$	$\mathrm{O}\left(\frac{CL_g}{\epsilon} \log \frac{1}{\epsilon}\right)$
総計算量（分散増大条件）	$\mathrm{O}\left(\frac{L_g(1+\beta)}{\epsilon} \log \frac{1}{\epsilon}\right)$	$\mathrm{O}\left(\frac{L_g(1+\beta)}{\epsilon} \log \frac{L_g}{\epsilon L_h}\right)$	$\mathrm{O}\left(CL_g(1+\beta)(\log \frac{1}{\epsilon})^2\right)$

式化の簡便さと，収束が効率的であることから，多くの分野で用いられてきた.

文献 [1] は，確率的近接 DC アルゴリズム (stochastic proximal DC algorithm, SPD) を提案している．提案手法は，関数値と勾配が確率的にしか観測できない確率的問題設定において有効な手法である．この設定での最適化手法はボルツマンマシンの学習など広い応用がある．さらに，Expectation-Maximization (EM) 法や Monte Carlo EM (MCEM) 法といった隠れ変数の構造を利用した手法は，SPD アルゴリズムの変種と捉えることができる．これらの手法は，DC 計画問題であると捉えることにより，SPD アルゴリズムによってさらに効率的な学習が可能になる.

表 1 は提案手法の計算量に関して，一般的設定（g のみが L_g-平滑），h が平滑な凸関数の場合（g, h がともに L_g, L_h-平滑），そして f が Polyak–Łojasiewicz 条件（2.3.3 節で詳述）を満たしている場合について比較したものである．表 1 の 2 行目は，部分問題に特に条件を課さない場合の全体計算量を表している．RSG [10] は Lipschitz 連続な平滑非凸目的関数を最適化するための確率的最適化手法であるが，これも提案手法と同等の $\mathrm{O}(L_g/\epsilon^2)$ なる計算量を達成することが（証明を少し修正することで）示せる．しかし，全体計算量は部分問題を解く際に前の反復の解から開始し十分小さなステップサイズを用いて最適化を行う warm-start といった技法を考慮に入れていないため，実応用における SPD の性能は表 1 に示した理論値よりも高くなることが実験的に確認されている.

2.2　確率的 DC アルゴリズム

ここでは，g と h の確率的勾配（真の勾配に観測ノイズが乗ったもの）しか観測できない状況を考える．ボルツマンマシンなど，多くの場合で g や h の勾配を計算するのに大規模な和や積分を計算する必要がある．その計算を省略するために，ランダムサンプリングで置き換えることを考える．確率的勾配のみが観測される状況はこのような設定に対応している．上記の問題設定で，SPD アルゴリズムをこれから説明する．H_k をサイズが $d \times d$ の正定値対称行列とし，$\|\cdot\|_{H_k}$ を H_k によって定義される Mahalanobis 距離とする：つまり，$v \in \mathbb{R}^d$ に対して，$\|v\|_{H_k} = \sqrt{\langle v, H_k v\rangle}$ とする．$v_h(x)$ を $\nabla h(x)$ の普遍推定量とし，σ_h^2 を v_h の分散の上界とする；$\mathbb{E}[v_h(x)] = \nabla h(x)$, $\mathbb{E}[\|v_h(x) - \nabla h(x)\|_2^2] \le \sigma_h^2$. x_k を k-反復目における暫定解とする.

x_k を更新して x_{k+1} を得るために，SPD は次で与えられる部分問題を確率的最適化によって近似的に解く：

$$SP(k) : \min_{x \in \mathbb{R}^d} \{\phi_k(x) \stackrel{\text{def}}{=} g(x) + \frac{1}{2}\|x - x_k\|_{H_k}^2 - (h(x_k) + \langle v_h(x_k), x - x_k\rangle)\}\,. \tag{2}$$

この更新式の近接勾配法との類似点に注意されたい．通常の決定的な DC アルゴリズムとこの更新式 (2) との違いは，後者は確率的な近似と近接項 $\frac{1}{2}\|x - x_k\|_{H_k}^2$ があることである．この近接項は x_{k+1} が前の値 x_k からノルム $\|\cdot\|_{H_k}$ の意味で遠く離れないように制御するための項である．実用的には H_k として，(i) $H_k = \mu I_d$, $\mu > 0$ や (ii) 2 回微分可能な h については $H_k = \left|\mathrm{diag}\left(\nabla^2 h(x_k)\right)\right| + \mu I_d$ を用いることが多い．なお，$|\cdot|$ は要素ごとに絶対値を取ることを表す．この部分問題 (2) を正確に解くのは非実用的であるため，部分問題の近似解に関して次のような条件を課す：

$$\mathbb{E}[\phi_k(x_{k+1}) | \mathcal{F}_k] \le \phi_k^* + \delta\,. \tag{3}$$

ただし，$\mathcal{F}_k$ は k 回目の反復までの履歴（より正確には増大情報系）で，ϕ_k^* は $SP(k)$ の最適値，そして $\delta > 0$ は部分問題の求解精度である．ここで，この部分問題を解く際に，前の反復の解から開始し十分小さなステップサイズを用いて最適化を行う warm-start を用いれば，実用上は容易に条件 (3) を満たすようにできる．SPD の具体的な手続きを Algorithm 1 に記述する.

Algorithm 1. SPD（確率的近接 DC アルゴリズム，Stochastic proximal DC algorithm）

Input: 初期値 x_1，反復回数の上界 M，$SP(k)$ を解くためのソルバー $\mathcal{A}$, $\mathcal{A}$ の内部反復回数 T.
$R \in \{1, 2, \ldots, M\}$ を一様ランダムに選択.
for $k = 1$ **to** $R - 1$ **do**
　H_k を更新.
　$\nabla h(x_k)$ の確率的近似である $v_h(x_k)$ を計算.
　$x_{k+1} \leftarrow \mathcal{A}$ を T 反復して $SP(k)$ を解いて得られた解.
end for
return x_R.

2.3 理論解析

本節ではSPDアルゴリズムの収束解析を与える．ここでは簡単のため，H_k として $\mu_k I_d$ のみを考える．まず最初に平滑性を以下のように定義する．

定義 1. 関数 ϕ がある $L_\phi > 0$ に対して $(L_\phi$-) 平滑であるとは，$\forall x, \forall y \in \mathbb{R}^d$ で

$$\|\nabla\phi(x) - \nabla\phi(y)\| \le L_\phi\|x-y\|_2,$$

を満たすことと定義する．

2.3.1 一般的設定

解のよさとして目的関数 f の勾配の二乗の期待値を用いた場合，提案手法によって得られる解のよさは次の定理のように評価することができる．

定理 2. g は L_g-平滑で，部分問題の解は期待値条件 (3) を満たし，f の最適解 f_* は下に有界であるとする．$\mu_k = \mathrm{O}(L_g)$ かつ，$\mu_k = \Omega(L_g)$ か $\sigma_h = 0$ のどちらかが成り立っているとする．すると，次が成り立つ：

$$\mathbb{E}[\|\nabla f(x_R)\|_2^2] \le \mathrm{O}\left(L_g\delta + \sigma_h^2 + \frac{L_g(f(x_1)-f_*)}{M}\right).$$

2.3.2 滑らかな h

本節では，h が平滑な場合の収束解析について述べる．計算複雑度を評価するにあたり，アルゴリズムを少し修正する：SPD の反復数 R を，$\{1,2,\ldots,M\}$ の代わりに $\{2,3,\ldots,M+1\}$ から一様ランダムに選択する（ただし，M は正の整数）．すると，次の収束定理を得る．この定理より，L_h が小さければ SPD はより速い収束を達成することがわかる．

定理 3. $L_h = \mathrm{O}(L_g)$ かつ，f の最適解 f_* は下に有界であるとする．すると，次が成り立つ：

$$\mathbb{E}[\|\nabla f(x_R)\|_2^2] \le \mathrm{O}\left(L_g\delta + \sigma_h^2 + \frac{L_h(f(x_1)-f_*)}{M}\right).$$

2.3.3 Polyak–Łojasiewicz 条件

ここでは，Polyak–Łojasiewicz 条件（PL 条件）のもと，SPD アルゴリズムを改良した二重ループ型 SPD (Algorithm 2) の収束解析を与える．なお，Polyak–Łojasiewicz 条件は以下で与えられる．

定義 4. 凸とは限らない関数 ϕ が Polyak–Łojasiewicz 条件（PL 条件）を満たすとは，ある正の定数 $C > 0$ が存在して，任意の $x \in \mathbb{R}^d$ において

$$\phi(x) - \min\phi \le C\|\nabla\phi(x)\|_2^2 \tag{4}$$

が成り立つことと定義する．

この仮定が成り立っていれば，関数の大きさが勾配の大きさで抑えられるため，勾配が 0 に近づくほど関数値が小さくなることが保証される．特に，強凸関数は PL 条件を満たすことが知られており，平滑性と合わせて勾配法が強凸関数の最小化において線形収束するために本質的な役割を果たす条件である．その意味で，PL 条件は強凸関数における重要な性質を取り出して非凸関数へ拡張したものとみなせる．

Algorithm 2. 二重ループ型 SPD

Input: 初期値 y_1, 外側ループの反復数 N, Algorithm 1 の引数 $M, \mathcal{A}, T$.
for $t = 1$ **to** $N-1$ **do**
  $y_{t+1} \leftarrow$ Algorithm 1 $(y_t, M, \mathcal{A}, T)$.
end for
return y_N.

Algorithm 1 と Algorithm 2 は実質的に最適化を進める途中のどの段階で解を返すかの違いしかないため, 実装上はほとんど修正の必要はない．$\delta = \mathrm{O}(\epsilon/L_g)$, $M = \mathrm{O}(CL_g/2)$ とし，$\sigma_h^2 = \mathrm{O}(\epsilon)$ とすると，定理 2 と式 (4) より，

$$\mathbb{E}[\|\nabla f(y_{t+1})\|_2^2] \le \epsilon + \frac{\mathbb{E}[\|\nabla f(y_t)\|_2^2]}{2}$$

であることが容易に確認できる．この再帰的関係式から，$\mathbb{E}[\|\nabla f(y_{t+1})\|_2^2] \le 2\epsilon + (\frac{1}{2})^t\|\nabla f(y_1)\|_2^2$ であることがすぐにわかる．これは，Algorithm 2 の外部反復を $N = \mathrm{O}(\log 1/\epsilon)$ 回実行することにより，誤差 ϵ の解が求まることを示している．よって，次の定理を得る．

定理 5. 定理 2 と同じ条件を仮定し，さらに目的関数 f は Polyak–Łojasiewicz 条件を満たしているとする．δ, M と σ_h を上記のように設定する．すると，SPD アルゴリズムの内部ループの計算量も含めた ϵ-解を求めるための総計算量は $\mathrm{O}(CL_g \log\frac{1}{\epsilon})$ で抑えられる．

これらの結果を総合して，各条件における全計算量を表 1 にまとめる．さらに，「分散増大条件」を追加で仮定することで計算量を改善させることができる [1] が，詳細は省く．ここで，表 1 の β はこの分散増大条件に関わる定数である．

3. 二重加速確率的分散縮小勾配降下法

本節では，文献 [2] で提案された，分散縮小法と呼ばれる手法に Nesterov の加速法を組み込んだ凸関数の確率的最適化手法を紹介する．機械学習では凸関数の有限和を最小化する問題が頻繁に現れ，そのような関数を最適化するために分散縮小技法を用いた加速確率的最適化手法が多く提案されている（加速確率的双対座標上昇法 (accelerated stochastic dual coordinate ascent, ASDCA) [11]，Universal Catalyst (UC) 法 [12]，加速近接勾配座標降下法 (accelerated proximal coordinate gradient，APCG) [13]，確率的主双対座標降下法 (stochastic primal-dual coordinate, SPDC) [14], 加速ミニバッチ近接確率的分散縮小勾配法 (accelerated mini-batch proximal stochastic variance reduced gradient, AccProxSVRG) [15, 16], Katyusha [17]）．

[2] で提案された手法は，二重加速確率的分散縮小勾配降下法 (doubly accelerated stochastic variance reduced dual averaging, DASVRDA) と呼ばれるものであり，従来手法と比べてミニバッチ法を有効活用できる手法である．なお，ミニバッチ法とは更新ごとに 1 個の観測点のみを用いるのではなく，複数個の観測点（ミニバッチ）を用いる方法である．DASVRDA はこのミニバッチのサイズに対する計算効率がよい手法である．DASVRDA の性質およびその各種既存手法との比較を表 2 にまとめる．

3.1 問題設定：正則化付き経験誤差最小化

この節では，問題設定および理論で重要な仮定を述べる．ここで考える最適化問題は以下の正則化付きの経験誤差最小化問題 (regularized empirical risk minimization, ERM) である：

$$\min_{x\in\mathbb{R}^d} \ \{P(x) \stackrel{\text{def}}{=} F(x) + R(x)\}. \tag{5}$$

ただし，$F(x) = \frac{1}{n}\sum_{i=1}^n f_i(x)$ である．ここで，各 $f_i : \mathbb{R}^d \to \mathbb{R}$ は L_i-平滑な凸関数で $R : \mathbb{R}^d \to \mathbb{R}$ は近接写像が容易に計算できるという意味で単純な凸関数であるとする．R は微分不可能でも構わない．この形をした最適化問題は，機械学習で頻繁に現れる基本的な問題である．たとえば，正則化付きロジスティック回帰は次のように定式化される：

$$\min_{x\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \log\{1+\exp(-b_i a_i^\top x)\} + R(x), \tag{6}$$

ただし各 $a_i \in \mathbb{R}^d$ は i 番目の観測の特徴ベクトルで，各 $b_i \in \{\pm 1\}$ はそれに対応する教師ラベルであり，また $R(x)$ は正則化関数である．正則化関数 $R(x)$ の例として，ℓ_1-正則化 $R(x) = \lambda\|x\|_1$ $(\lambda \geq 0)$ やエラスティックネット正則化 $R(x) = \lambda_1\|x\|_1 + (\lambda_2/2)\|x\|_2^2$ $(\lambda_1, \lambda_2 \geq 0)$ などがある．

ここで，目的関数に次の仮定を置く．

仮定 1.

1. 最適化問題 (5) には最適解 x_* が存在する．
2. 各 f_i は凸関数で，L_i-平滑である．
3. 正則化関数 R は凸で，以下で定義される近接写像が $\mathrm{O}(d)$ の計算量で計算できる：

$$\mathrm{prox}_R(y) = \arg\min_{x\in\mathbb{R}^d}\left\{\frac{1}{2}\|x-y\|^2 + R(x)\right\}.$$

仮定 1 に加えて強凸性を満たす目的関数に対するアルゴリズムも考察する．

仮定 2. ある $\mu > 0$ が存在して，目的関数 P が（最適解の周りにおいて）μ-一点強凸関数である．つまり，P は唯一の最適解 $x^* = \arg\min_{x\in\mathbb{R}^d} f(x)$ をもち，

$$\frac{\mu}{2}\|x - x_*\|^2 \leq P(x) - P(x_*) \ (\forall x \in \mathbb{R}^d),$$

を満たす．

一点強凸性の条件は通常の強凸性 [19] に比べて弱い条件であることに注意されたい．

3.2 アルゴリズムの詳細

Algorithm 3. DASVRDA$^{\mathrm{ns}}(\widetilde{x}_0, \gamma, \{L_i\}_{i=1}^n, m, b, S)$

$\widetilde{x}_{-1} = \widetilde{z}_0 = \widetilde{x}_0$, $\widetilde{\theta}_0 = 1 - \frac{1}{\gamma}$, $\bar{L} = \frac{1}{n}\sum_{i=1}^n L_i$, $Q = \{q_i\} = \left\{\frac{L_i}{n\bar{L}}\right\}$, $\eta = \frac{1}{\left(1+\frac{\gamma(m+1)}{b}\right)\bar{L}}$.

for $s = 1$ to S **do**

$\widetilde{\theta}_s = \left(1 - \frac{1}{\gamma}\right)\frac{s+2}{2}$, $\widetilde{y}_s = \widetilde{x}_{s-1} + \frac{\widetilde{\theta}_{s-1}-1}{\widetilde{\theta}_s}(\widetilde{x}_{s-1} - \widetilde{x}_{s-2}) + \frac{\widetilde{\theta}_{s-1}}{\widetilde{\theta}_s}(\widetilde{z}_{s-1} - \widetilde{x}_{s-1})$.

$(\widetilde{x}_s, \widetilde{z}_s) = \mathrm{AccSVRDA}(\widetilde{y}_s, \widetilde{x}_{s-1}, \eta, m, b, Q)$.

end for

return $\widetilde{x}_S$.

表 2 提案手法 DASVRDA と SVRG (SVRG^{++} [18]), ASDCA (UC), APCG, SPDC, Katyusha, AccProxSVRG との比較

	μ-strongly convex			Non-strongly convex		
	Total computational cost in size b mini-batch settings	Necessary size of mini-batches		Total computational cost in size b mini-batch settings	Necessary size of mini-batches	
		$L/\mu \geq n$	$L/\mu \leq n$		$\frac{L}{\varepsilon} \geq n\log^2(n)$	$\frac{L}{\varepsilon} \leq n\log^2(n)$
SVRG (SVRG^{++})	$\mathrm{O}\left(d\left(n+\frac{bL}{\mu}\right)\log\left(\frac{1}{\varepsilon}\right)\right)$	Unattainable	Unattainable	$\mathrm{O}\left(d\left(n\log\left(\frac{1}{\varepsilon}\right)+\frac{bL}{\varepsilon}\right)\right)$	Unattainable	Unattainable
ASDCA (UC)	$\widetilde{\mathrm{O}}\left(d\left(n+\sqrt{\frac{nbL}{\mu}}\right)\log\left(\frac{1}{\varepsilon}\right)\right)$	Unattainable	Unattainable	$\widetilde{\mathrm{O}}\left(d\left(\frac{n+\sqrt{nbL}}{\sqrt{\varepsilon}}\right)\right)$	Unattainable	Unattainable
APCG	$\mathrm{O}\left(d\left(n+\sqrt{\frac{nbL}{\mu}}\right)\log\left(\frac{1}{\varepsilon}\right)\right)$	$\mathrm{O}(n)$	$\mathrm{O}(n)$	No direct analysis	Unattainable	Unattainable
SPDC	$\mathrm{O}\left(d\left(n+\sqrt{\frac{nbL}{\mu}}\right)\log\left(\frac{1}{\varepsilon}\right)\right)$	$\mathrm{O}(n)$	$\mathrm{O}(n)$	No direct analysis	Unattainable	Unattainable
Katyusha	$\mathrm{O}\left(d\left(n+\sqrt{\frac{nbL}{\mu}}\right)\log\left(\frac{1}{\varepsilon}\right)\right)$	$\mathrm{O}(n)$	$\mathrm{O}(n)$	$\mathrm{O}\left(d\left(n\log\left(\frac{1}{\varepsilon}\right)+\sqrt{\frac{nbL}{\varepsilon}}\right)\right)$	$\mathrm{O}(n)$	$\mathrm{O}(n)$
AccProxSVRG	$\mathrm{O}\left(d\left(n+\left(\frac{n-b}{n-1}\right)\frac{L}{\mu}+b\sqrt{\frac{L}{\mu}}\right)\log\left(\frac{1}{\varepsilon}\right)\right)$	$\mathrm{O}\left(\sqrt{\frac{L}{\mu}}\right)$	$\mathrm{O}\left(n\sqrt{\frac{\mu}{L}}\right)$	No direct analysis	Unattainable	Unattainable
DASVRDA	$\mathrm{O}\left(d\left(n+\sqrt{\frac{nL}{\mu}}+b\sqrt{\frac{L}{\mu}}\right)\log\left(\frac{1}{\varepsilon}\right)\right)$	$\mathrm{O}\left(\sqrt{n}\right)$	$\mathrm{O}\left(n\sqrt{\frac{\mu}{L}}\right)$	$\mathrm{O}\left(d\left(n\log(\frac{n}{b})+\sqrt{\frac{nL}{\varepsilon}}+b\sqrt{\frac{L}{\varepsilon}}\right)\right)$	$\mathrm{O}\left(\sqrt{n}\right)$	$\tilde{\mathrm{O}}\left(n\sqrt{\frac{\varepsilon}{L}}\right)$

n は目的関数を構成する有限和の個数，d は変数の次元，b はミニバッチサイズ，L は平滑性パラメータ，μ は目的関数の強凸性パラメータ，ε は解の精度である．“Necessary size of mini-batches” は最適な反復回数（強凸関数は $\mathrm{O}(\sqrt{L/\mu}\log(1/\varepsilon))$，非強凸関数は $\mathrm{O}(\sqrt{L/\varepsilon})$）を達成するために必要なミニバッチサイズである．ミニバッチサイズが b の場合，全データを用いた勾配の計算は n/b の計算量としている．“Unattainable” はアルゴリズムがミニバッチサイズを n にしても，最適な反復回数を達成しないことを意味する．$\widetilde{\mathrm{O}}$ は log-多項式オーダーを隠したオーダー表記である．

Algorithm 4. AccSVRDA $(\widetilde{y}, \widetilde{x}, \eta, m, b, Q)$

$x_0 = z_0 = \widetilde{y}$, $\bar{g}_0 = 0$, $\theta_0 = \frac{1}{2}$.
for $k = 1$ to m **do**
 $i_k^1, \ldots, i_k^b \sim Q$ を独立同一に生成し，$I_k = \{i_k^\ell\}_{\ell=1}^b$ とする．
 $\theta_k = \frac{k+1}{2}$, $y_k = \left(1 - \frac{1}{\theta_k}\right) x_{k-1} + \frac{1}{\theta_k} z_{k-1}$.
 $g_k = \frac{1}{b} \sum_{i \in I_k} \frac{1}{nq_i} \left(\nabla f_i(y_k) - \nabla f_i(\widetilde{x})\right) + \nabla F(\widetilde{x})$.
 $\bar{g}_k = \left(1 - \frac{1}{\theta_k}\right) \bar{g}_{k-1} + \frac{1}{\theta_k} g_k$.
 $z_k = \mathrm{prox}_{\eta\theta_k\theta_{k-1}R}\left(z_0 - \eta\theta_k\theta_{k-1}\bar{g}_k\right)$.
 $x_k = \left(1 - \frac{1}{\theta_k}\right) x_{k-1} + \frac{1}{\theta_k} z_k$.
end for
return (x_m, z_m).

Algorithm 5. DASVRDA$^{\mathrm{sc}}(\check{x}_0, \gamma, \{L_i\}_{i=1}^n, m, b, S, T)$

for $t = 1$ to T **do**
 $\check{x}_t = \mathrm{DASVRDA}^{\mathrm{ns}}(\check{x}_{t-1}, \gamma, \{L_i\}_{i=1}^n, m, b, S)$.
end for
return $\check{x}_T$.

本節では，提案アルゴリズムの具体的な手続きの詳細を述べる．非強凸な目的関数に対する DASVRDA の手続きを Algorithm 3 に記述する．DASVRDA のモーメンタム（慣性）ステップは通常の加速法とは少し異なる：通常はモーメンタム項 $((\widetilde{\theta}_{s-1}-1)/\widetilde{\theta}_s)(\widetilde{x}_{s-1}-\widetilde{x}_{s-2})$ を現在の解 $\widetilde{x}_{s-1}$ に加えるだけであるが，DASVRDA ではさらに「積極的な解」$\widetilde{z}_{s-1}$ を用意し，これを用いて $(\widetilde{\theta}_{s-1}/\widetilde{\theta}_s)(\widetilde{z}_{s-1} - \widetilde{x}_{s-1})$ も加える．

次に，内部ループである Accelerated SVRDA (Algorithm 4) に移る．Algorithm 4 は，基本的に加速正則化双対平均加法 (accelerated stochastic regularized dual averaging, AccSDA) と分散縮小勾配法を組み合わせたものである．この内部ループにおいては z_k をこれまでの分散縮小した勾配 $\bar{g}_k$ の平均を用いて更新する．通常の分散縮小勾配法では現在の分散縮小勾配 $\bar{g}_k$ のみを用いるが，その平均を用いる点が双対平均加法の特徴的な点である．こうすることにより，遅延更新と呼ばれる疎データに対する高速な更新が可能になり総計算量を抑えることが可能になる（詳細は文献 [2] を参照されたい）．

Algorithm 5 は，目的関数が一点強凸性を満たすときの手順である．強凸関数で通常用いられる定数モーメンタム項を用いた加速 [19] を行うのではなく，Algorithm 3 ではリスタート法と呼ばれる手法を用いる．リスタート法は理論的にも実用的にも利点がある．まず，リスタート法は目的関数が強凸関数である必要はなく，一点強凸関数で十分である．通常の定数モーメンタム項を用いる場合は目的関数は通常の意味での強凸関数である必要がある．さらに，リスタート法を採用することによって，「適応的リスタート法」[20] を使うことができる．これは，強凸性パラメータ μ を事前に設定する必要はなく，アルゴリズムが適応的にリスタートのタイミングを調整する方法である．ヒューリスティクスではあるが，経験的に非常に有効な方法であることが知られている．

3.3 DASVRDA 法の収束解析

この節では，DASVRDA の収束解析を与える．まず，非強凸目的関数に対する DASVRDA$^{\mathrm{ns}}$ の収束解析を考察する．

定理 6. 仮定 1 が成り立っているとする．$\widetilde{x}_0 \in \mathbb{R}^d$, $\gamma \geq 3$, $m \in \mathbb{N}$, $b \in [n]$ および $S \in \mathbb{N}$ とする．すると，DASVRDA$^{\mathrm{ns}}(\widetilde{x}_0, \gamma, \{L_i\}_{i=1}^n, m, b, S)$ は次を満たす：

$$
\mathbb{E}[P(\widetilde{x}_S)-P(x_*)] \le \frac{4}{(S+2)^2}(P(\widetilde{x}_0)-P(x_*)) + \frac{8\left(1+\frac{\gamma(m+1)}{b}\right)\bar{L}}{\left(1-\frac{1}{\gamma}\right)^2(S+2)^2 m(m+1)}\|\widetilde{x}_0-x_*\|^2.
$$

この上界を最小にする最適な γ は $\gamma=(3+\sqrt{9+8b/(m+1)})/2=\mathrm{O}(1+b/m)$ で与えられることがわかる．この値を γ_* とする．すると，定理 6 より，次の系が得られる．

系 7. 仮定 1 が満たされているとする．$\widetilde{x}_0\in\mathbb{R}^d$, $\gamma=\gamma_*$, $m\propto n/b$ かつ $b\in[n]$ とする．もし，$S=\mathrm{O}(1+\sqrt{(P(\widetilde{x}_0)-P(x_*))/\varepsilon}+(1/m+1/\sqrt{mb})\sqrt{\bar{L}\|\widetilde{x}_0-x_*\|^2/\varepsilon})$ と設定すると，$\mathbb{E}[P(\widetilde{x}_S)-P(x_*)]\le\varepsilon$ を満たすまでの $\mathrm{DASVRDA^{ns}}(\widetilde{x}_0,\gamma_*,\{L_i\}_{i=1}^n,m,b,S)$ の総計算量は

$$
\mathrm{O}\left(d\left(n\sqrt{\frac{P(\widetilde{x}_0)-P(x_*)}{\varepsilon}}+(b+\sqrt{n})\sqrt{\frac{\bar{L}\|\widetilde{x}_0-x_*\|^2}{\varepsilon}}\right)\right)
$$

で抑えられる．

注釈 8. 系 7 より，非強凸な目的関数における DASVRDA の総計算量は $\mathrm{O}(d(n/\sqrt{\varepsilon}+(b+\sqrt{n})\sqrt{\bar{L}/\varepsilon}))$ で抑えられる．しかし，さらに初期化を工夫することで，$\mathrm{O}(d(n\log(n/b)+(b+\sqrt{n})\sqrt{\bar{L}/\varepsilon}))$ に減らすことができる．詳細は文献 [2] を参照されたい．

次に，一点強凸目的関数に対する $\mathrm{DASVRDA^{sc}}$ アルゴリズムを考察する．定理 6 を一点強凸目的関数に適用することで次の定理を得る．これより，$\mathrm{DASVRDA^{sc}}$ は線形収束することがわかる．

定理 9. 仮定 1 と仮定 2 が成り立っているとする．$\check{x}_0\in\mathbb{R}^d$, $\gamma=\gamma_*$, $m\in\mathbb{N}$, $b\in[n]$ かつ $T\in\mathbb{N}$ とする．$\rho\overset{\mathrm{def}}{=}4/(S+2)^2+16(1+\gamma_*(m+1)/b)\bar{L}/\{(1-1/\gamma_*)^2(m+1)m\mu(S+2)^2\}$ とする．もし，S が十分に大きく $\rho\in(0,1)$ が成り立つなら，$\mathrm{DASVRDA^{sc}}(\check{x}_0,\gamma_*,\{L_i\}_{i=1}^n,m,b,S,T)$ は，以下の収束を達成する：

$$
\mathbb{E}[P(\check{x}_T)-P(x_*)]\le\rho^T[P(\check{x}_0)-P(x_*)].
$$

定理 9 より次の系を得る．

系 10. 仮定 1 と仮定 2 が満たされているとする．$\check{x}_0\in\mathbb{R}^d$, $\gamma=\gamma_*$, $m\propto n/b$, $b\in[n]$ とする．ある S が存在して，$S=\mathrm{O}(1+(b/n+1/\sqrt{n})\sqrt{\bar{L}/\mu})$ かつ $1/\log(1/\rho)=\mathrm{O}(1)$ とすることができる．さらに，もし $T=\mathrm{O}(\log(P(\check{x}_0)-P(x_*)/\varepsilon)$ とすれば，$\mathrm{DASVRDA^{sc}}(\check{x}_0,\gamma_*,\{L_i\}_{i=1}^n,m,b,S,T)$ の ε-解を得るまでの総計算量は，

$$
\mathrm{O}\left(d\left(n+(b+\sqrt{n})\sqrt{\frac{\bar{L}}{\mu}}\right)\log\left(\frac{P(\check{x}_0)-P(x_*)}{\varepsilon}\right)\right)
$$

で抑えられる．

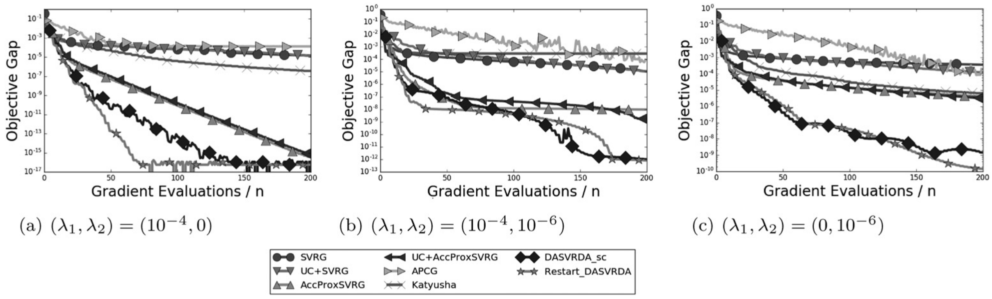
注釈 11. 系 10 から，ミニバッチサイズ b が $\mathrm{O}(\sqrt{n})$ であれば，$\mathrm{DASVRDA^{sc}}(\check{x}_0,\gamma_*,\{L_i\}_{i=1}^n,n/b,b,S,T)$ は総計算量を $\mathrm{O}(d(n+\sqrt{n\bar{L}/\mu})\log(1/\varepsilon))$ のままに抑えることができる．一方で，APCG, SPDC および Katyusha は $\mathrm{O}(d(n+\sqrt{nb\bar{L}/\mu})\log(1/\varepsilon))$ かかってしまい，これらより計算量を削減できていることがわかる[1]．

注釈 12. さらに，系 10 から，$L/\mu\ge n$ のとき，最適な反復回数 $\mathrm{O}(\sqrt{L/\mu}\log(1/\varepsilon))$ を達成するために $\mathrm{DASVRDA^{sc}}$ は $\mathrm{O}(\sqrt{n})$ のミニバッチサイズで十分であることを示唆している．一方，APCG や SPDC, Katyusha といった既存手法は $\mathrm{O}(n)$ のミニバッチサイズが必要で，AccProxSVRG は $\mathrm{O}(\sqrt{L/\mu})$ のミニバッチサイズが必要である．さらに，$L/\mu\le n$ のとき，われわれの手法は $\mathrm{O}(n\sqrt{\mu/L})$ のミニバッチサイズで十分である．

3.4 数値実験

本節では，DASVRDA とその他の代表的な既存手法との比較を行う．比較手法としては以下を採用した：SVRG [22] (and $\mathrm{SVRG^{++}}$ [18]), AccProx-SVRG [15], Universal Catalyst [12], APCG [13] および Katyusha [17]．実験では，二値判別に対する正則化ロジスティック回帰問題（式 (6)）を扱い，正則化項としてエラスティックネット正則化 $\lambda_1\|\cdot\|_1+(\lambda_2/2)\|\cdot\|_2^2$ を用いた．ここでは，データとして a9a dataset を用いた結果のみを示す．正則化パラメータとしては，$(\lambda_1,\lambda_2)=(10^{-4},0)$, $(10^{-4},10^{-6})$, $(0,10^{-6})$ の三種類の組合せを用いた．一番最初の設定では目的関数は非強凸であり，残りの二つの設定では目的関数は強凸である．

[1] なお，論文 [2] が出版された後，Katyusha の改良版が提案され，同じ計算量を達成することが示されている [21].



(a) $(\lambda_1, \lambda_2) = (10^{-4}, 0)$　(b) $(\lambda_1, \lambda_2) = (10^{-4}, 10^{-6})$　(c) $(\lambda_1, \lambda_2) = (0, 10^{-6})$

図 1　a9a データセットにおける比較
左から順に正則化パラメータを $(\lambda_1, \lambda_2) = (10^{-4}, 0), (10^{-4}, 10^{-6}), (0, 10^{-6})$ に設定.

図 1 に各種手法の比較を示す．縦軸の "Objective Gap" は $P(x) - P(x_*)$ を意味し，横軸の "Gradient Evaluations $/n$" は，確率的勾配 ∇f_i を評価した回数を n で割ったものである．"Restart_DASVRDA" は DASVRDA に適応的リスタート法を適用したものである．全体として，DASVRDA および Restart_DASVRDA 法は既存手法を大きく改善していることが見て取れる．興味深いことに，適応的リスタート法を用いたDASVRDA は，非強凸関数に対しても局所的な強凸性を捉えることで通常の DASVRDA 法よりも速い収束を示している．

4.　まとめと今後の課題

本稿では，勾配を用いた二つの確率的最適化手法を紹介した．前半では確率的 DC 計画法を，後半では二重加速確率的分散縮小勾配降下法を紹介した．いずれの手法も確率的勾配を用いることで計算量を減らし，全体として効率的な最適化を実現している．機械学習では大規模データを扱う必要があり，そのような需要に確率的勾配を用いた一次法はよく当てはまっている．現在は深層学習の流行もあり，非凸関数の最適化に対する確率的勾配降下法が大きな注目を集めている．しかし，深層学習は目的関数の形状や性質がまだよくわかっておらず，深層学習の学習理論も考慮に入れたより効率的な確率的最適化手法の開発が望まれている．

参考文献

[1] A. Nitanda and T. Suzuki, "Stochastic difference of convex algorithm and its application to training deep Boltzmann machines," In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 470–478, 2017.
[2] T. Murata and T. Suzuki, "Doubly accelerated stochastic variance reduced dual averaging method for regularized empirical risk minimization," *Advances in Neural Information Processing Systems*, pp. 608–617, 2017.
[3] 鈴木大慈，『確率的最適化（機械学習プロフェッショナルシリーズ）』，講談社，2015.
[4] T. P. Dinh and E. B. Souad, "Algorithms for solving a class of nonconvex optimization problems: Methods of subgradient," *North-Holland Mathematics Studies*, **129**, pp. 249–271, 1986.
[5] A. Argyriou, R. Hauser, C. A. Micchelli and M. Pontil, "A DC-programming algorithm for kernel selection," In *Proceedings of the 23rd International Conference on Machine Learning*, pp. 41–48, 2006.
[6] H. A. L. Thi, L. H. Minh, N. V. Vinh and T. P. Dinh, "A DC programming approach for feature selection in support vector machines learning," *Advances in Data Analysis and Classification*, **2**, pp. 259–278, 2008.
[7] A. Ferrer, "Representation of a polynomial function as a difference of convex polynomials, with an application," *Lectures Notes in Economics and Mathematical Systems*, **502**, pp. 189–207, 2001.
[8] S. Wang, A. Schwing and R. Urtasun, "Efficient inference of continuous Markov random fields with polynomial potentials," *Advances in Neural Information Processing Systems*, **25**, pp. 936–944. 2014.
[9] A. A. Ahmadi and G. Hall, "DC decomposition of nonconvex polynomials with algebraic techniques," *Mathematical Programming*, **169**, pp. 69–94, 2018.
[10] S. Ghadimi and G. Lan, "Stochastic first- and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, **23**, pp. 2341–2368, 2013.
[11] S. Shalev-Shwartz and T. Zhang, "Stochastic dual coordinate ascent methods for regularized loss," *The Journal of Machine Learning Research*, **14**, pp. 567–599, 2013.
[12] H. Lin, J. Mairal and Z. Harchaoui, "A universal catalyst for first-order optimization," *Advances in Neural Information Processing Systems*, pp. 3384–3392, 2015.
[13] Q. Lin, Z. Lu and L. Xiao, "An accelerated proximal coordinate gradient method," *Advances in Neural Information Processing Systems*, pp. 3059–3067, 2014.
[14] Y. Zhang and X. Lin, "Stochastic primal-dual coordinate method for regularized empirical risk minimization," In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 353–361, 2015.
[15] A. Nitanda, "Stochastic proximal gradient descent with acceleration techniques," *Advances in Neural Information Processing Systems*, pp. 1574–1582, 2014.

[16] A. Nitanda, "Accelerated stochastic gradient descent for minimizing finite sums," In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pp. 195–203, 2016.
[17] Z. Allen-Zhu, "Katyusha: The first direct acceleration of stochastic gradient methods," In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1200–1205, 2017.
[18] Z. Allen-Zhu and Y. Yuan, "Improved SVRG for non-strongly-convex or sum-of-non-convex objectives," In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 1080–1089, 2016.
[19] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, Applied Optimization Series 87, Springer Science & Business Media, 2013.
[20] B. O'Donoghue and E. Candes, "Adaptive restart for accelerated gradient schemes," *Foundations of Computational Mathematics*, **15**, pp. 715–732, 2015.
[21] Z. Allen-Zhu, "Katyusha: The first direct acceleration of stochastic gradient methods," *Journal of Machine Learning Research*, **18**(221), pp. 1–51, 2018.
[22] L. Xiao and T. Zhang, "A proximal stochastic gradient method with progressive variance reduction," *SIAM Journal on Optimization*, **24**, pp. 2057–2075, 2014.