第 5 回とめ研究所若手研究者懸賞論文 2024

連続最適化アルゴリズムの探索方向ノイズと深層学習モデルの汎化性能の関係

明治大学大学院 理工学研究科 情報科学専攻
佐藤 尚樹
指導教員：飯塚 秀明

近年の人工知能の急速な技術革新は，深層学習をはじめとする機械学習手法の進歩によるものである．一般に深層学習モデルの訓練には，確率的勾配降下法 (SGD) やモーメンタム法 (SGD with momentum) をはじめとする連続最適化アルゴリズムが使用され，それらの収束性と安定性については，理論と応用の両側面から非常に活発に研究されてきた．しかし，未だに理論と応用の間には大きな乖離があり，実験的には観察されているが理論的には解明されていない現象が多く存在する．例えば，SGD with momentum は特にバッチサイズが大きい時に SGD よりも汎化性能が優れることが観察されているが，それがなぜかは明らかになっていない．このような，いつ，どのような条件で，なぜ訓練が成功するのかという当然の疑問には，アルゴリズムの収束性と安定性に関する既存の理論だけでは答えることができない．

本稿は，最も基本的な連続最適化アルゴリズムである SGD と SGD with momentum に注目し，それらが暗黙のうちに抱えている確率的なノイズに着目する．まず，このノイズが目的関数を平滑化しているとみなせること，その平滑化の度合いは訓練に使われるアルゴリズムの学習率やバッチサイズ等のパラメータで定まることを示す．次に，アルゴリズムごとの平滑化の度合いを推定することで，確率的ノイズの大きさと深層学習モデルの汎化性能との関係について理論的に解析する．得られた理論は畳み込みニューラルネットワークベースの深層学習モデルによる画像分類タスクの数値実験によって検証する．最後に，これらの結果によって上述したいくつかの未解明の現象の説明が可能となることを示す．

なお本稿は，arXiv にて公表済みで，査読中につき未採録の以下の 2 つの論文 [1, 2] の内容の一部を再構成したものである．また，論文 [1, 2] は日本オペレーションズ・リサーチ学会研究発表会にて発表済み [3, 4] である．

[1] Naoki Sato, and Hideaki Iiduka. Using stochastic gradient descent to smooth nonconvex functions: Analysis of implicit graduated optimization with optimal noise scheduling. https://arxiv.org/abs/2311.08745, 2023.

[2] Naoki Sato, and Hideaki Iiduka. Role of momentum in smoothing objective function and generalizability of deep neural networks. https://arxiv.org/abs/2402.02325, 2024.

[3] 佐藤尚樹，飯塚秀明．確率的勾配降下法の平滑化効果を利用した段階的最適化手法によるディープニューラルネットワークの大域的最適化．日本オペレーションズ・リサーチ学会 2024 年春季研究発表会, 筑波大学 筑波キャンパス 春日エリア, 2024.

[4] 佐藤尚樹，飯塚秀明．目的関数の平滑化とディープニューラルネットワークの汎化性能におけるモーメンタム法の慣性項の役割．日本オペレーションズ・リサーチ学会 2024 年秋季研究発表会, 南山大学, 2024.

目次

第 1 章 はじめに

1.1 背景

1.1.1 深層学習における連続最適化

機械学習とは，与えられたデータから得られるルールやパターンを機械が学習することで，未知のデータに対して分類や予測をする技術のことである．機械学習手法の一つである深層学習モデルの訓練においては，訓練データ $\boldsymbol{z}_i (i = 1, 2, \cdots, n)$ と深層学習モデルのパラメータ $\boldsymbol{x} \in \mathbb{R}^d$ に対して，モデルが出力した予測値と訓練データの正解値との誤差 $f_i(\boldsymbol{x})$ の平均を目的関数とし，これを最小にするようなパラメータを探索する．すなわち，以下のように定義される最適化問題を解くことを，モデルを訓練するという．

$$\text{Minimize} \quad f(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} f_i(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{x} \in \mathbb{R}^d. \tag{1}$$

問題 (1) の目的関数 f は，訓練データという限られたデータから計算されるため経験損失関数といい，問題 (1) を経験損失最小化問題という．一般に経験損失関数は微分可能な非凸関数となる．ここで，非凸関数とは，2 つ以上の局所的最適解を有する関数のことである．問題 (1) の最適解 $\boldsymbol{x}^\star \in \mathbb{R}^d$ が，訓練データに対して最も適切な，深層学習モデルのパラメータである．したがって，モデルの訓練を成功させるためには，何らかの方法で問題 (1) を解き，最適解 $\boldsymbol{x}^\star \in \mathbb{R}^d$ を近似する必要がある．

1.1.2 経験損失最小化問題のための連続最適化アルゴリズム

問題 (1) の解法には，勾配法と呼ばれる，関数の微分情報である勾配を利用する最適化アルゴリズムがよく使用される．勾配法とは，ベクトル $\boldsymbol{x}_t \in \mathbb{R}^d$ を現在点とするとき，$\eta_t > 0$ の大きさで $\boldsymbol{d}_t$ ベクトル方向へ進むことで次点 $\boldsymbol{x}_{t+1}$ へ更新する手法の総称である．すなわち，漸化式

$$\boldsymbol{x}_{t+1} := \boldsymbol{x}_t + \eta_t \boldsymbol{d}_t$$

によって点列 $(\boldsymbol{x}_t)_{t \in \mathbb{N}} \in \mathbb{R}^d$ を生成し，時刻または反復回数 t を十分大きくしたとき，$\boldsymbol{x}_t$ が問題 (1) の最適解 $\boldsymbol{x}^\star \in \mathbb{R}^d$ を近似できるようにする．ここで，$\eta_t > 0$ を時刻 t での学習率と呼び，ベクトル $\boldsymbol{d}_t$ を時刻 t での探索方向と呼ぶ．最も単純な勾配法は，探索方向に目的関数 f の全勾配 $\nabla f(\boldsymbol{x}_t)$ を利用する最急降下法：

$$\boldsymbol{x}_{t+1} := \boldsymbol{x}_t - \eta_t \nabla f(\boldsymbol{x}_t)$$

である．しかし，深層学習モデルのパラメータの総数 d は数十万から数兆に及び，訓練データの総数 n は数万から数千万に及ぶため，各時刻で全勾配 $\nabla f(\boldsymbol{x}_t)$ の計算を必要とする最急降下法による訓練は現実的ではない．次のように定義される確率的勾配降下法 (Stochastic Gradient Descnet; SGD)[5] は，各時刻で n 個全ての訓練データを使う代わりに，ランダムに選ばれた b $(\leq n)$ 個の訓練データ $\mathcal{S}_t$ に対して計算されたミニバッチ確率的勾配 $\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)$ を探索方向に利用するため，バッチサイズ b を適切に設定すれば実行することができる：

$$\boldsymbol{x}_{t+1} := \boldsymbol{x}_t - \eta_t \nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t).$$

SGD は深層学習モデルの訓練に使用される最適化アルゴリズムの中で最も単純なアルゴリズムであり，これまでに多くの改良手法が提案されている．モーメンタム法 (SGD with momentum)[6, 7, 8] は，時刻 t の探索方向 $\boldsymbol{m}_t$ にひとつ前の探索方向 $\boldsymbol{m}_{t-1}$ の情報を活用することで，収束を加速させる手法である：

$$\begin{aligned}\boldsymbol{m}_t &:= \nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) + \beta \boldsymbol{m}_{t-1} \\ \boldsymbol{x}_{t+1} &:= \boldsymbol{x}_t - \eta_t \boldsymbol{m}_t.\end{aligned}$$

ベクトル $\boldsymbol{m}_{t-1}$ を慣性項と呼び，$\beta \in [0, 1)$ を慣性係数と呼ぶ．$\beta = 0$ での SGD with momentum は SGD と一致する．SGD with momentum は SGD よりも収束が速く，しかも汎化性能も優れることが知られている．

1.1.3 経験損失関数と汎化性能

機械学習において最も重視されているのはモデルの汎化性能である．汎化性能とは，訓練に使われたデータ以外のデータ，いわゆるテストデータに対しても正しく予測，分類する能力のことである．問題 (1) に SGD 等の最適化アルゴリズムを適用すると，経験損失関数値 $f(\boldsymbol{x})$ を極めて小さくすることのできる近似解を手に入れることができる．このとき，訓練データに対する予測精度である訓練精度は 100% に近くなるが，テストデータに対する予測精度であるテスト精度は 100% にはならず，訓練データセットとモデルの種類によって 60%〜90% ほどとなるのが一般的である．

テスト精度を 100% にすることが機械学習の究極の目標だが，理論的に扱うことができるのは経験損失関数のみであるため，経験損失関数と汎化性能の関係について多くの先行研究がある．その多くは経験損失関数の最適解の周りの関数の形状に関するものであり，「その近傍が平坦な最適解は，その近傍が急峻な最適解よりも優れた汎化性をもたらす」という仮説が主流である．図 1 に示すように，一般に訓練データのみから計算される経験損

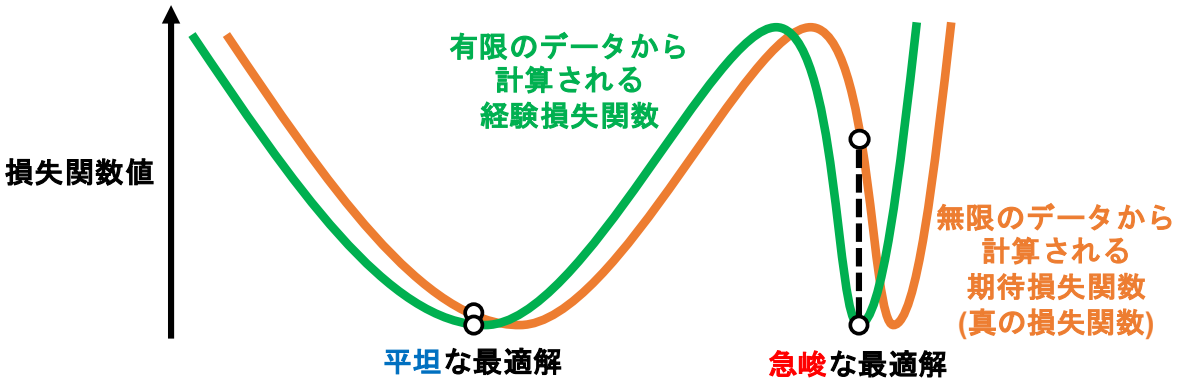

図 1　経験損失関数の形状の概念図．高い汎化性能を得るためには期待損失関数の最小化が必要であるが，我々が扱えるのは経験損失関数のみである．よって，最適化アルゴリズムが収束する経験損失関数の最適解付近において，2 つの関数の誤差 (点線部) がより少ない，平坦な最適解が高い汎化性能をもたらすと考えられている．

失関数と，テストデータを含む無限のデータから計算される期待損失関数の間には誤差があるはずである．期待損失関数値を最小にすることができれば，テスト精度は 100% となるはずであるから，我々が最適化できるのは経験損失関数である一方，期待損失関数値も考慮しなければならない．2 つの損失関数の誤差を考慮すると，経験損失関数の最適解のうち，その近傍が平坦な最適解の方が，低い期待損失関数値を達成できるはずである (図 1)．このような仮説の源流は，1997 年の Hochreiter と Schmidhuber による研究 [9] であり，この仮説に関する先行研究は多い．例えば，複数の先行研究が近似解の近傍での関数の滑らかさの指標として “Sharpness” を提案しており [10, 11, 12, 13, 14]，Sharpness とモデルの汎化性能には相関があることが実験的に示されている [10, 15, 16, 17]．なお，本稿の主張は上述の仮説を支持し，それを裏付けるものである (詳細は 5.3 節を参照)．

1.1.4　SGD の確率的ノイズによる平滑化

全ての訓練データを 1 度に扱う最急降下法 (GD) と比べて，SGD は 1 度に b 個のデータしか扱わないため，各反復でどのデータが選ばれるかによって確率的なノイズが発生している．このノイズは，各時刻におけるそれぞれの探索方向の誤差 $\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t) =: \boldsymbol{\omega}_t^{\mathrm{SGD}}$ で表現するのが自然であろう．いくつかの先行研究によって，本稿が考察する畳み込みニューラルネットワーク (CNN) ベースの深層学習モデルによる画像分類タスクにおいては，ノイズ $\boldsymbol{\omega}_t^{\mathrm{SGD}}$ は正規分布に従うことが実験的に観察されている [18, 19]．

非凸関数の最適化においては，SGD は GD よりも汎化性能の優れる解に収束することが知られている [20, 21]．Kleinberg らは，以下のような議論から，その原因が SGD の確

率的ノイズにある可能性を示唆した [22].

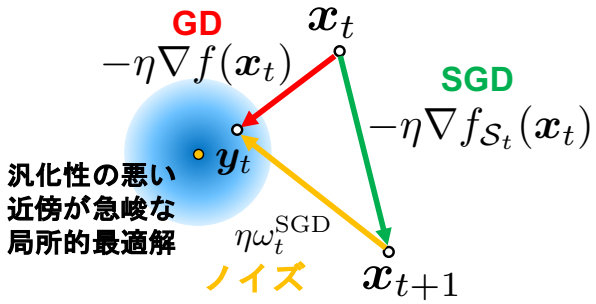

図 2　SGD の確率的ノイズによる局所的最適解の回避の概念図．青い領域の中心が，汎化性能の優れない近傍が急峻な局所的最適解だとすると，全勾配 (赤色の矢印) はまっすぐにその局所解を目指す．一方，SGD はノイズ $\boldsymbol{\omega}_t^{\mathrm{SGD}}$(黄色の矢印) があるおかげで，その局所解を回避することができる (緑色の矢印).

時刻 t における深層学習モデルのパラメータを $\boldsymbol{x}_t$ とする．このとき，$\boldsymbol{x}_t$ から GD で更新したパラメータを $\boldsymbol{y}_t$，SGD で更新したパラメータを $\boldsymbol{x}_{t+1}$ とする，すなわち，

$$\begin{aligned}\boldsymbol{y}_t &:= \boldsymbol{x}_t - \eta\nabla f(\boldsymbol{x}_t)\\ \boldsymbol{x}_{t+1} &:= \boldsymbol{x}_t - \eta\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)\\ &= \boldsymbol{x}_t - \eta\left(\nabla f(\boldsymbol{x}_t) + \boldsymbol{\omega}_t^{\mathrm{SGD}}\right)\end{aligned}$$

とする (図 2 も参照)．すると，

$$\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[\boldsymbol{y}_{t+1}] = \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[\boldsymbol{y}_t] - \eta\nabla\underbrace{\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[f\left(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SGD}}\right)\right]}_{=:\hat{f}(\boldsymbol{y}_t)} \tag{2}$$

が成り立つ (式 (2) の導出は付録 A を参照)．ここで，新たに関数 $\hat{f}(\boldsymbol{y}_t) := \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[f(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SGD}})]$ を定義すると，関数 $\hat{f}$ は正規分布に従う確率変数 $\boldsymbol{\omega}_t^{\mathrm{SGD}}$ で関数 f を畳み込んだ形をしているため，関数 f をある程度平滑化した関数が関数 $\hat{f}$ であるといえる (定義 2.1 と文献 [23] を参照)．さらに，

$$\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[\boldsymbol{y}_{t+1}] = \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[\boldsymbol{y}_t] - \eta\nabla\hat{f}(\boldsymbol{y}_t)$$

が成り立つことから，パラメータ $\boldsymbol{y}_t$ は，期待値の意味では GD で更新されているとみなせる．すなわち，関数 f を SGD で最適化することと，関数 $\hat{f}$ を GD で最適化することは期待値の意味では等価であるとみなせる．したがって，最適化に SGD を使うというだけで，その確率的ノイズ $\boldsymbol{\omega}_t^{\mathrm{SGD}}$ によって関数 $f(\boldsymbol{x}_t)$ はある程度平滑化されているとみなせる．それによって急峻な局所的最適解が消滅するため，点列がそこに陥ることがなく，GD よりも高い汎化性能をもたらすといえる．

1.1.5 SGD with momentum の確率的ノイズと汎化性能の矛盾

一般に，SGD with momentum は，慣性項を加えることによって SGD の確率的ノイズを削減していると考えられている [24, 25]．一方，1.1.4 節より，確率的ノイズが関数の平滑化をもたらし，結果的に優れた汎化性能をもたらすといえる．そして，SGD with momentum の方が，特にバッチサイズが大きいときに SGD よりも高い汎化性をもたらすことが観察されている．これら 3 つの主張が全て正しければ，ここには矛盾が生じている．つまり，慣性項は SGD の確率的ノイズを削減しているため，SGD with momentum の汎化性能は SGD よりも悪くなるはずだが，そうはなっていないという点に矛盾がある．

1.2 動機

本研究は以下の 2 つに動機づけられている．

1. 式 (2) は，SGD の確率的ノイズが関数を平滑化していることを示唆しているが，その平滑化の度合いはどの程度か，またそれを決定する要因は何かは明らかになっていない．本稿はこれらを明らかにすることを目指す．
2. SGD の確率的ノイズによる平滑化の議論を SGD with momentum に拡張し，同様にその平滑化の度合いが何によって定まるかを明らかにすることを目指す．

1.3 貢献

上述の動機の下で，次のような成果を得ることができた．

1. SGD と SGD with momentum(SHB と NSHB) の確率的ノイズによる平滑化の度合いは，学習率 η，バッチサイズ b，慣性係数 β，確率的勾配の分散 C_{opt}^2，勾配ノルムの上界 K_{opt}^2 を用いて，次のように表されることを示した (第 3 章)．

$$\delta^{\mathrm{SGD}} = \eta\sqrt{\frac{C_{\mathrm{SGD}}^2}{b}},$$
$$\delta^{\mathrm{SHB}} = \eta\sqrt{\left(1+\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}\right)\frac{C_{\mathrm{SHB}}^2}{b}+\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}K_{\mathrm{SHB}}^2},$$
$$\delta^{\mathrm{NSHB}} = \eta\sqrt{\frac{1}{1-\beta}\frac{C_{\mathrm{NSHB}}^2}{b}},$$

   なお，C_{opt}^2 と K_{opt}^2 の厳密な定義は，2.2 節を参照されたい．
2. 最適化アルゴリズムの収束解析と，最適なバッチサイズの推定の議論を介して，推

定が困難な確率的勾配の分散 C_{opt}^2 を推定することに成功した (第 4 章).

3. 最適化アルゴリズムごとの平滑化の度合いを数値で導出することで，平滑化の度合いと深層学習モデルの汎化性能には相関があり，大きすぎず小さすぎない平滑化の度合いが優れた汎化性能をもたらすことを示した (第 5 章 5.1 節).
4. 平滑化の度合いは Sharpness と同様に関数の滑らかさ/鋭さを表す指標であることを示し，Sharpness と汎化性能の間には相関がない一方で，平滑化の度合いと汎化性能の間には相関があることを示した (第 5 章 5.3 節).
5. 平滑化の度合いに関する議論から，慣性項を加えることで削減される確率的ノイズと，汎化性能に寄与する確率的ノイズが別物であることを示すことで，1.1.5 節で述べた SGD with momentum の確率的ノイズと汎化性能に関する矛盾を解決した (第 5 章 5.2 節).

第 2 章 数学的準備

2.1 記法と定義

0 と自然数全体からなる集合を $\mathbb{N}$ で表し，任意の $m \in \mathbb{N} \setminus \{0\}$ に対して，$[m] := \{1, 2, \ldots, m\}$ とする．内積 $\langle \cdot, \cdot \rangle$ によって導出されたノルム $\| \cdot \|$ をもつ d 次元ユークリッド空間を $\mathbb{R}^d$ で表す．d 次元の単位行列を $I_d \in \mathbb{R}^{d \times d}$ で表す．平均 $\boldsymbol{\mu} \in \mathbb{R}^d$，分散 $\Sigma \in \mathbb{R}^{d \times d}$ の正規分布を $\mathcal{N}(\boldsymbol{\mu}; \Sigma)$ で表す．d 個のパラメータ $\boldsymbol{x} \in \mathbb{R}^d$ をもつ深層学習モデルと，n 個の訓練データ $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_n\}$ に対して，$f(\boldsymbol{x}) := \frac{1}{n} \sum_{i \in [n]} f_i(\boldsymbol{x})$ とする．ただし，$f_i(\boldsymbol{x})$ は $\boldsymbol{x} \in \mathbb{R}^d$ と i 番目の訓練データ $\boldsymbol{z}_i$ に対して計算される損失関数である．ξ を $\boldsymbol{x} \in \mathbb{R}^d$ と独立な確率変数とし，ξ に関する確率変数 X の期待値を $\mathbb{E}_\xi[X]$ で表す．時刻 t における i 個目の訓練データのサンプリングによって生成される確率変数を $\xi_{t,i}$ とし，$\boldsymbol{\xi}_t := (\xi_{t,1}, \xi_{t,2}, \ldots, \xi_{t,b})$ は点列 $(\boldsymbol{x}_k)_{k=0}^t \subset \mathbb{R}^d$ と独立である．ここで，b $(\leq n)$ はバッチサイズである．$\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \ldots,$ は独立であるから，全期待値 $\mathbb{E}$ を $\mathbb{E} := \mathbb{E}_{\boldsymbol{\xi}_0} \mathbb{E}_{\boldsymbol{\xi}_1} \cdots \mathbb{E}_{\boldsymbol{\xi}_t}$ と定義できる．時刻 t における任意の $\boldsymbol{x} \in \mathbb{R}^d$ に対して，$f(\cdot)$ の確率的勾配を $\mathsf{G}_{\boldsymbol{\xi}_t}(\boldsymbol{x})$ と表す．時刻 t における b 個のサンプルをミニバッチと呼び，$\mathcal{S}_t$ で表す．$\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)$ は $\mathcal{S}_t$ に対する関数 $f(\boldsymbol{x}_t)$ のミニバッチ確率的勾配であり，b 個の確率的勾配の平均で定義される．すなわち，$\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) := \frac{1}{b} \sum_{i \in [b]} \mathsf{G}_{\xi_{t,i}}(\boldsymbol{x}_t)$ である．

2.2 仮定

本稿は次の仮定の下で解析を行う．

仮定 2.1 (A1) $f_i\colon \mathbb{R}^d \to \mathbb{R}$ $(i \in [n])$ は連続的微分可能かつ L_f-リプシッツ関数である．
すなわち，任意の $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ に対して，$|f(\boldsymbol{x}) - f(\boldsymbol{y})| \leq L_f \|\boldsymbol{x} - \boldsymbol{y}\|$ が成り立つ．
(A2) $(\boldsymbol{x}_t)_{t\in\mathbb{N}} \subset \mathbb{R}^d$ を最適化アルゴリズムにより生成された点列とする．このとき，
(i) 任意の $t \in \mathbb{N}$ に対して，次の式が成り立つ．

$$\mathbb{E}_{\boldsymbol{\xi}_t}\left[\mathsf{G}_{\boldsymbol{\xi}_t}(\boldsymbol{x}_t)\right] = \nabla f(\boldsymbol{x}_t).$$

(ii) 任意の $t \in \mathbb{N}$ に対して，次の式を満たす非負定数 C_{opt}^2 が存在する．

$$\mathbb{E}_{\boldsymbol{\xi}_t}\left[\|\mathsf{G}_{\boldsymbol{\xi}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)\|^2\right] \leq C_{\mathrm{opt}}^2.$$

(A3) 任意の $t \in \mathbb{N}$ に対して，全勾配 ∇f はミニバッチ $\mathcal{S}_t$ で次のように近似される．

$$\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) := \frac{1}{b}\sum_{i\in[b]} \mathsf{G}_{\xi_{t,i}}(\boldsymbol{x}_t) = \frac{1}{b}\sum_{\{i\colon \boldsymbol{z}_i\in\mathcal{S}_t\}} \nabla f_i(\boldsymbol{x}_t).$$

(A4) 任意の $t \in \mathbb{N}$ に対して，次の式を満たす正の定数 K_{opt} が存在する．

$$\mathbb{E}\left[\|\nabla f(\boldsymbol{x}_t)\|^2\right] \leq K_{\mathrm{opt}}^2.$$

ここで (A2)(ii) において，C_{opt}^2 とは，最適化アルゴリズムごとに定義される定数であり，添え字の “opt” は最適化アルゴリズムの名前を表す．例えば，点列 $(\boldsymbol{x}_t)_{t\in\mathbb{N}}$ が SGD によって生成された点列であるときは，$\mathbb{E}_{\boldsymbol{\xi}_t}\left[\|\mathsf{G}_{\boldsymbol{\xi}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)\|^2\right] \leq C_{\mathrm{SGD}}^2$ を満たす C_{SGD}^2 が存在することを仮定している．(A4) の定数 K_{opt} も同様である．なお，定数 C_{opt}^2 は，その定義から，確率的勾配 $\mathsf{G}_{\boldsymbol{\xi}_t}(\boldsymbol{x}_t)$ の分散を表していることが分かる．一般に，これらの定数，特に確率的勾配の分散 C_{opt}^2 は最適化アルゴリズムに依存しない値として仮定されることが多いが，その定義から明らかなように，定数 C_{opt}^2 は確率変数 $\boldsymbol{\xi}_t$ だけでなくパラメータ $\boldsymbol{x}_t$ にも依存する値である．任意の時刻 $t \in \mathbb{N}$ において，異なる最適化アルゴリズムが異なるパラメータ $\boldsymbol{x}_t$ へと導くことは自明であるから，この仮定は妥当であることを補足しておこう．

2.3 アルゴリズム

本稿は，3 つの最適化アルゴリズムを考察する．

SGD は最急降下法の確率的変種であり，探索方向に全勾配 $\nabla f(\boldsymbol{x}_t)$ を利用する最急降下法に対して，SGD はミニバッチ確率的勾配 $\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) := \frac{1}{b}\sum_{i\in[b]} \mathsf{G}_{\xi_{t,i}}$ を利用する．学習率 η とバッチサイズ b はユーザーが自由に設定できるパラメータであり，機械学習分野では慣例的に，バッチサイズには 2 の累乗の数が使用される．なお，文献によっては

アルゴリズム 2.1 確率的勾配降下法 (Stochastic Gradient Descent, SGD) [5]

Input: $\boldsymbol{x}_0 \in \mathbb{R}^d, \eta > 0, b \in [n]$
  for $t = 0$ to $T-1$ **do**
    $\boldsymbol{x}_{t+1} := \boldsymbol{x}_t - \eta \nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)$
  end for
  return $\boldsymbol{x}_T$

"SGD" が後述の SGD with momentum を指す，または含む場合があるが，本項における "SGD" は SGD without momentum のことを指す．

SGD with momentum は，SGD の改良手法であり，時刻 t の探索方向に時刻 $t-1$ の探索方向の情報 (慣性項) を取り入れることで，最適解への収束を加速させる．多くの先行研究によって，Nesterov の加速勾配法 [26, 27, 28, 29] やロバストモーメンタム [30] など，慣性項のとり方が微妙に異なる SGD with momentum がいくつも提案されている．本稿はその中で最も基本的な，以下の 2 つの SGD with momentum に焦点を当てる．なお，以下のアルゴリズムの名称は文献によって異なるため，本稿における呼称は SGD with momentum の代表的な先行研究である文献 [31] に倣っている．

アルゴリズム 2.2 Stochastic Heavy Ball (SHB) [6]

Input: $\boldsymbol{x}_0 \in \mathbb{R}^d, \eta > 0, \beta \in [0,1), b \in [n], \boldsymbol{m}_{-1} := \boldsymbol{0}$
  for $t = 0$ to $T-1$ **do**
    $\boldsymbol{m}_t := \nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) + \beta \boldsymbol{m}_{t-1}$
    $\boldsymbol{x}_{t+1} := \boldsymbol{x}_t - \eta \boldsymbol{m}_t$
  end for
  return $\boldsymbol{x}_T$

アルゴリズム 2.3 Normalized Stochastic Heavy Ball (NSHB) [7]

Input: $\boldsymbol{x}_0 \in \mathbb{R}^d, \eta > 0, \beta \in [0,1), b \in [n], \boldsymbol{d}_{-1} := \boldsymbol{0}$
  for $t = 0$ to $T-1$ **do**
    $\boldsymbol{d}_t := (1-\beta)\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) + \beta \boldsymbol{d}_{t-1}$
    $\boldsymbol{x}_{t+1} := \boldsymbol{x}_t - \eta \boldsymbol{d}_t$
  end for
  return $\boldsymbol{x}_T$

SHB と NSHB の唯一の違いは，探索方向に凸結合をとるかとらないか，という点である．SHB は探索方向 $\boldsymbol{m}_t$ に凸結合をとらず，NSHB は探索方向 $\boldsymbol{d}_t$ に凸結合をとっている．ここで，m 個のベクトル $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^m \in \mathbb{R}^d$ に対して，$\sum_{i=1}^m \alpha_i = 1, \alpha_i \geq 0$ $(i = 1, \ldots, m)$ を満たす実数 $\alpha_1, \ldots, \alpha_m$ を用いて，

$$\boldsymbol{x} = \alpha_1 \boldsymbol{x}^1 + \cdots + \alpha_m \boldsymbol{x}^m$$

と表されるベクトル $\boldsymbol{x} \in \mathbb{R}^d$ を $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^m$ の凸結合という．凸結合は理論的に扱いやすい性質として知られており，解析が比較的容易であるため，NSHB を理論的に解析する先行研究は多い．実際，多くの先行研究で SGD with momentum (SGDM) と紹介されているアルゴリズムは NSHB であり，特にその収束は既に十分に解析されているが，SHB の収束を理論的に解析する先行研究は少ない．対照的に，数値実験では NSHB の性能は SGD とほぼ変わらないことが観察されており，ほとんど利用されないが，SHB は収束率と汎化性能ともに SGD よりも優れた性能を発揮することが観察されており，非常によく利用される．実際，PyTorch [32] や Tensorflow [33] 等の一般的な機械学習ライブラリが提供する SGD with momentum は SHB である．

すなわち，SHB は理論的には解析されないが実験ではよく利用され，NSHB は理論的にはよく解析されるが実験では利用されない．なぜ実験で NSHB よりも SHB の方が優れているのかは明らかになっていないが，本稿がその理由を明らかにする．

2.4 勾配ノイズと探索方向ノイズ

最適化アルゴリズムがもつ確率的なノイズに注目する先行研究は多い [22, 18, 34, 19]．これまでの全ての先行研究が考慮してきた確率的ノイズとは，最適化アルゴリズムの種類によらず

$$\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)$$

で表され，勾配ノイズと呼ばれる．そもそも SGD をはじめとする確率的最適化アルゴリズムが確率的ノイズをもっているのは，各時刻 t で全てのデータを一度に扱う GD と比べて，1 度に b 個のデータ $\mathcal{S}_t$ しか扱わないためであるから，この定式化はもっともらしい．それに対して，本稿が平滑化の議論のために考慮する確率的ノイズは，ある最適化アルゴリズムの探索方向と GD の探索方向，すなわち全勾配 $\nabla f(\boldsymbol{x}_t)$ の誤差で定義される．した

がって，このノイズは最適化アルゴリズムごとに次のように定義される．

$$\begin{aligned}
&(\text{SGD の探索方向ノイズ})\ \boldsymbol{\omega}_t^{\mathrm{SGD}} := \nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t),\\
&(\text{SHB の探索方向ノイズ})\ \boldsymbol{\omega}_t^{\mathrm{SHB}} := \boldsymbol{m}_t - \nabla f(\boldsymbol{x}_t),\\
&(\text{NSHB の探索方向ノイズ})\ \boldsymbol{\omega}_t^{\mathrm{NSHB}} := \boldsymbol{d}_t - \nabla f(\boldsymbol{x}_t).
\end{aligned}$$

本稿ではこのノイズを探索方向ノイズと呼び，$\boldsymbol{\omega}_t^{\mathrm{opt}}$ で表す．なお，SGD だけは勾配ノイズと探索方向ノイズが一致している．勾配ノイズと探索方向ノイズの定式化の意味とその役割については，5.2 節で述べる．

2.5 関数の平滑化

一般に，ある関数 f の平滑化は，正規分布に従う確率変数で関数 f を畳み込むことで達成される [23].

定義 2.1 任意の関数 $f : \mathbb{R}^d \to \mathbb{R}$ を平滑化することで得られる関数 $\hat{f}_\delta : \mathbb{R}^d \to \mathbb{R}$ を

$$\hat{f}_\delta(\boldsymbol{x}) := \mathbb{E}_{\boldsymbol{u}\sim\mathcal{N}\left(\boldsymbol{0};\frac{1}{\sqrt{d}}I_d\right)}\left[f(\boldsymbol{x}-\delta\boldsymbol{u})\right]$$

で表す．ここで，$\boldsymbol{u} \in \mathbb{R}^d$ は正規分布に従う確率変数で，$\delta > 0$ は平滑化の度合いを表している．

ある関数を定義 2.1 に従って平滑化するとき，δ が大きければ大きいほど関数はより平滑化される．当然，$\delta = 0$ のとき，関数 $\hat{f}_\delta$ は元の関数 f に一致する．

次の補題は，平滑化された関数 $\hat{f}_\delta$ の重要な性質を示すものである．

補題 2.1 関数 $\hat{f}_\delta$ を関数 f を平滑化して得られた関数だとする．このとき, 任意の $\boldsymbol{x} \in \mathbb{R}^d$ に対して,

$$\left|\hat{f}_\delta(\boldsymbol{x}) - f(\boldsymbol{x})\right| \leq \delta L_f$$

が成り立つ.

(証明) 定義 2.1 と仮定 (A1) から，任意の $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ に対して,

$$\begin{aligned}
\left|\hat{f}_\delta(\boldsymbol{x}) - f(\boldsymbol{x})\right| &= \left|\mathbb{E}_{\boldsymbol{u}}\left[f(\boldsymbol{x}-\delta\boldsymbol{u})\right] - f(\boldsymbol{x})\right|\\
&= \left|\mathbb{E}_{\boldsymbol{u}}\left[f(\boldsymbol{x}-\delta\boldsymbol{u}) - f(\boldsymbol{x})\right]\right|\\
&\leq \mathbb{E}_{\boldsymbol{u}}\left[\left|f(\boldsymbol{x}-\delta\boldsymbol{u}) - f(\boldsymbol{x})\right|\right]\\
&\leq \mathbb{E}_{\boldsymbol{u}}\left[L_f\|(\boldsymbol{x}-\delta\boldsymbol{u}) - \boldsymbol{x}\|\right]\\
&= \mathbb{E}_{\boldsymbol{u}}\left[L_f\delta\|\boldsymbol{u}\|\right] = \delta L_f \mathbb{E}_{\boldsymbol{u}}\left[\|\boldsymbol{u}\|\right] \leq \delta L_f,
\end{aligned}$$

とできる． **(証明おわり)**

より平坦な近傍を有する最適解がより良い汎化をもたらすという仮説から，目的関数を平滑化することで汎化性能を向上させようとするとき，十分な大きさの平滑化の度合い δ が必要であろう．一方で，補題 2.1 は，平滑化の度合い δ が大きければ大きいほど，関数値の意味で関数 $\hat{f}_\delta$ と元の関数 f の差が大きくなっていくことを示している．したがって，平滑化の度合いには大きすぎず小さすぎない最適な値が存在すると考察できる．この詳細は 5.1 節で述べる．

最後に，関数の平滑化の例を 1 つ示そう．図 3 は，式 (3) のように定義される 1 変数の Rastrigin's 関数 [35, 36] と，定義 2.1 に従って計算された，その平滑化バージョンをプロットしたものである．

$$\text{（Rastrigin's 関数）}\quad f(x) := x^2 - 10\cos(2\pi x) + 10 \tag{3}$$

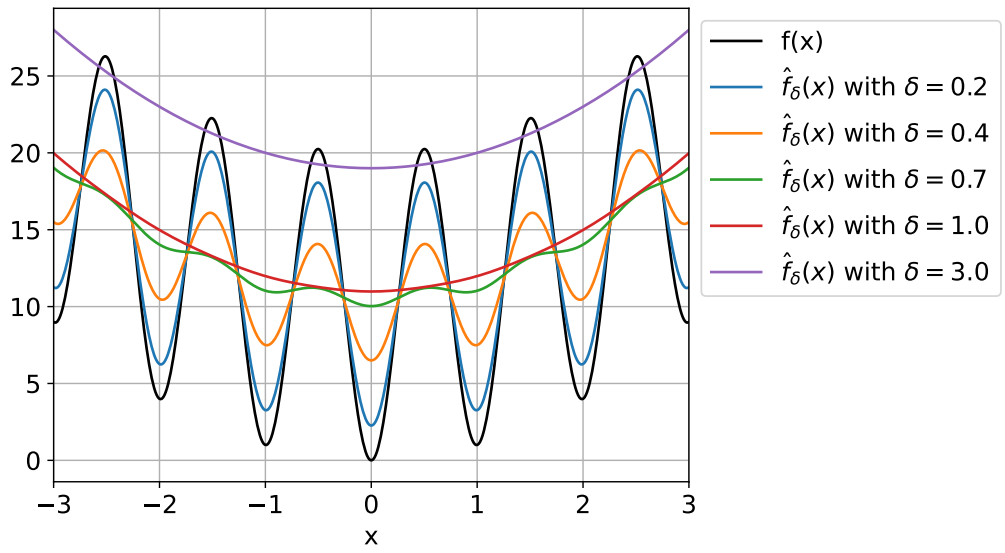

図 3　元の目的関数 f(式 (3) を参照) と，平滑化の度合い $\delta \in \{0.2, 0.4, 0.7, 1.0, 3.0\}$ を使って計算された $\hat{f}_\delta$.

δ が大きくなるほど関数がより平滑化され，徐々に関数の谷が削れていく様子が確認できる．$\delta = 1.0$ の段階で関数 $\hat{f}_\delta$ は完全に凸関数になっている．また，補題 2.1 が示す通り，δ が大きくなればなるほど，元の関数 $f(\boldsymbol{x})$ との距離が大きくなっていく様子も確認できる．

第 3 章 探索方向ノイズによる目的関数の平滑化

本章では，1.1.4 節の議論を進展させて，最適化アルゴリズムが暗黙のうちに抱えている確率的なノイズによる目的関数の平滑化の度合いが，学習率等のパラメータによって定まることを示す．

3.1 SGD の探索方向ノイズによる平滑化の度合い

1.1.4 節では，時刻 t における深層学習モデルのパラメータを $\boldsymbol{x}_t$ とし，$\boldsymbol{x}_t$ から GD で更新したパラメータを $\boldsymbol{y}_t$，SGD で更新したパラメータを $\boldsymbol{x}_{t+1}$ とすると，

$$\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[\boldsymbol{y}_{t+1}] = \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[\boldsymbol{y}_t] - \eta\nabla \underbrace{\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[f\left(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SGD}}\right)]}_{=:\hat{f}(\boldsymbol{y}_t)} \tag{4}$$

が成り立つことを紹介した．関数の平滑化の定義 (定義 2.1) から，関数 $\hat{f}$ は関数 f をある程度平滑化した関数であることが分かるが，その平滑化の度合いは未知である．これを明らかにするために，ノイズ $\boldsymbol{\omega}_t^{\mathrm{SGD}}$ に注目する．まずは次の定理 3.1 を紹介しよう．

定理 3.1 任意の $t \in \mathbb{N}$ に対して，仮定 (A2)(ii) と仮定 (A3) が成り立つとすると，

$$\mathbb{E}_{\boldsymbol{\xi}_t}\left[\left\|\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)\right\|^2\right] \leq \frac{C_{\mathrm{opt}}^2}{b}. \tag{5}$$

が成り立つ．

(証明) $t \in \mathbb{N}$，$\boldsymbol{\xi}_t := (\xi_{t,1}, \cdots, \xi_{t,b})^\top$ とすると，仮定 (A3) より，

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\xi}_t}\left[\left\|\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)\right\|^2 \middle| \boldsymbol{x}_t\right] &= \mathbb{E}_{\boldsymbol{\xi}_t}\left[\left\|\frac{1}{b}\sum_{i=1}^{b}\mathsf{G}_{\xi_{t,i}}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)\right\|^2\right] \\
&= \mathbb{E}_{\boldsymbol{\xi}_t}\left[\left\|\frac{1}{b}\sum_{i=1}^{b}\mathsf{G}_{\xi_{t,i}}(\boldsymbol{x}_t) - \frac{1}{b}\sum_{i=1}^{b}\nabla f(\boldsymbol{x}_t)\right\|^2\right] \\
&= \mathbb{E}_{\boldsymbol{\xi}_t}\left[\left\|\frac{1}{b}\sum_{i=1}^{b}\left(\mathsf{G}_{\xi_{t,i}}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)\right)\right\|^2\right] \\
&= \frac{1}{b^2}\mathbb{E}_{\boldsymbol{\xi}_t}\left[\left\|\sum_{i=1}^{b}\left(\mathsf{G}_{\xi_{t,i}}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)\right)\right\|^2\right]
\end{aligned}$$

が成り立つ．したがって，仮定 (A2)(ii) より，

$$\mathbb{E}_{\boldsymbol{\xi}_t}\left[\|\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)-\nabla f(\boldsymbol{x}_t)\|^2\middle|\boldsymbol{x}_t\right]=\frac{1}{b^2}\mathbb{E}_{\boldsymbol{\xi}_t}\left[\sum_{i=1}^{b}\left\|\mathsf{G}_{\xi_{t,i}}(\boldsymbol{x}_t)-\nabla f(\boldsymbol{x}_t)\right\|^2\right]\leq\frac{C_{\mathrm{opt}}^2}{b}.$$

とできる． **(証明おわり)**

定理 3.1 は任意の最適化アルゴリズムが生成する点列 $(\boldsymbol{x}_t)_{t\in\mathbb{N}}$ に対して成り立つものであることに注意する．$\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)-\nabla f(\boldsymbol{x}_t)$ は，任意の最適化アルゴリズムの勾配ノイズそのものであるから，定理 3.1 より，アルゴリズムの勾配ノイズとは，アルゴリズムごとの確率的勾配の分散 C_{opt}^2 であるといえる．

さて，$(\boldsymbol{x}_t)_{t\in\mathbb{N}}$ が SGD によって生成された点列であるとき，$\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)-\nabla f(\boldsymbol{x}_t)=:\boldsymbol{\omega}_t^{\mathrm{SGD}}$ が成り立つことと定理 3.1 から，特に SGD が生成した点列 $(\boldsymbol{x}_t)_{t\in\mathbb{N}}$ に対して，

$$\mathbb{E}_{\boldsymbol{\xi}_t}\left[\left\|\boldsymbol{\omega}_t^{\mathrm{SGD}}\right\|^2\right]\leq\frac{C_{\mathrm{SGD}}^2}{b}. \tag{6}$$

が成り立つ．$\boldsymbol{\omega}_t^{\mathrm{SGD}}$ が正規分布に従うという実験的観察から，正規分布に従う確率変数 $\boldsymbol{u}_t\in\mathbb{R}^d$ を使って，式 (6) を満たす $\boldsymbol{\omega}_t^{\mathrm{SGD}}$ は例えば次のように表すことができる．

$$\boldsymbol{\omega}_t^{\mathrm{SGD}}:=\sqrt{\frac{C_{\mathrm{SGD}}^2}{b}}\boldsymbol{u}_t\ \left(\boldsymbol{u}_t\sim\mathcal{N}\left(\boldsymbol{0};\frac{1}{\sqrt{d}}I_d\right)\right).$$

これを式 (4) に代入すると，定義 2.1 から，

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[\boldsymbol{y}_{t+1}\right]&=\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[\boldsymbol{y}_t\right]-\eta\nabla\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[f\left(\boldsymbol{y}_t-\eta\boldsymbol{\omega}_t^{\mathrm{SGD}}\right)\right]\\&=\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[\boldsymbol{y}_t\right]-\eta\nabla\mathbb{E}_{\boldsymbol{u}_t\sim\mathcal{N}\left(\boldsymbol{0};\frac{1}{\sqrt{d}}I_d\right)}\left[f\left(\boldsymbol{y}_t-\eta\sqrt{\frac{C_{\mathrm{SGD}}^2}{b}}\boldsymbol{u}_t\right)\right]\\&=\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[\boldsymbol{y}_t\right]-\eta\nabla\hat{f}_{\eta\sqrt{\frac{C_{\mathrm{SGD}}^2}{b}}}(\boldsymbol{y}_t),\end{aligned} \tag{7}$$

が成り立つ．この式は，関数 $\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[f\left(\boldsymbol{y}_t-\eta\boldsymbol{\omega}_t^{\mathrm{SGD}}\right)\right]$ は，関数 f を大きさ $\eta\sqrt{\frac{C_{\mathrm{SGD}}^2}{b}}=:\delta^{\mathrm{SGD}}$ のノイズで平滑化した関数であることを示している．さらに，式 (7) は，関数 $\hat{f}_{\delta^{\mathrm{SGD}}}$ のパラメータ $\boldsymbol{y}_t$ は期待値の意味では GD によって更新されていることを意味している．したがって，モデルのパラメータの最適化に SGD を使うというだけで，暗黙のうちに目的関数 f は SGD の探索方向ノイズ $\boldsymbol{\omega}_t^{\mathrm{SGD}}$ によって平滑化されているとみなせて，その平滑化の度合いは，学習率 η，バッチサイズ b と確率的勾配の分散 C_{SGD}^2 によって定まる．

関数 f は平滑化の度合い δ^{SGD} が大きければ大きいほど平滑化されるため，学習率 η は大きければ大きいほど，バッチサイズ b は小さければ小さいほど関数 f はより平滑化されることになる．元の関数 f の急峻な局所的最適解を回避するためには，ある程度大きな平滑化の度合いが必要であると考えられる．よって，例えば SGD を最適化に使うとき，大きすぎるバッチサイズを設定すると，平滑化の度合いは小さくなり，関数 f の平滑化が十分でないため点列は急峻な局所的最適解に陥りやすくなり，結果的にモデルの汎化性能は優れないと考察できる．

3.2 SGD with momentum の探索方向ノイズによる平滑化の度合い

全ての訓練データを 1 度に扱う GD と比べて，SGD with momentum もまた 1 度に b 個のデータしか扱わないため，各反復でどのデータが選ばれるかによって確率的なノイズ，すなわち探索方向ノイズが発生していると考えられる．3.1 節では，SGD の探索方向ノイズによる平滑化の度合いが $\delta^{\mathrm{SGD}} = \eta\sqrt{\dfrac{C_{\mathrm{SGD}}^2}{b}}$ によって定まることを示した．では，SGD に慣性項が追加された SGD with momentum の探索方向ノイズによる平滑化の度合いはどのようになるだろうか．本節では，3.1 節の SGD に対する議論を，SHB と NSHB の 2 つの SGD with momentum に拡張する．SGD with momentum のアルゴリズムについては 2.3 節を，その探索方向ノイズについては 2.4 節を参照されたい．

時刻 t における深層学習モデルのパラメータを $\boldsymbol{x}_t$ とし，$\boldsymbol{x}_t$ から GD で更新したパラメータを $\boldsymbol{y}_t$，SHB で更新したパラメータを $\boldsymbol{x}_{t+1}$ とする，すなわち，

$$\begin{aligned}\boldsymbol{y}_t &:= \boldsymbol{x}_t - \eta\nabla f(\boldsymbol{x}_t)\\ \boldsymbol{x}_{t+1} &:= \boldsymbol{x}_t - \eta\boldsymbol{m}_t\\ &= \boldsymbol{x}_t - \eta\left(\nabla f(\boldsymbol{x}_t) + \boldsymbol{\omega}_t^{\mathrm{SHB}}\right)\end{aligned}$$

とする．すると，SGD に対する議論と同様に，

$$\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SHB}}}[\boldsymbol{y}_{t+1}] = \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SHB}}}[\boldsymbol{y}_t] - \eta\nabla\underbrace{\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SHB}}}[f\left(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SHB}}\right)]}_{=:\hat{f}(\boldsymbol{y}_t)} \tag{8}$$

が成り立ち，関数の平滑化の定義 (定義 2.1) から，関数 $\hat{f}$ は関数 f をある程度平滑化した関数であることが分かる．NSHB に対しても同様である．それでは，肝心の平滑化の度合いがどのようになるか，探索方向ノイズ $\boldsymbol{\omega}_t^{\mathrm{SHB}}$ と $\boldsymbol{\omega}_t^{\mathrm{NSHB}}$ の大きさに注目して導出していく．まずは次の定理 3.2 を紹介する．なお，定理 3.2 の証明は，文献 [2] の Appendix C.2 章を参照されたい．

定理 3.2 任意の $t \in \mathbb{N}$ に対して，仮定 (A2)(ii)，(A3)，(A4) が成り立つとすると，

$$\mathbb{E}\left[\left\|\boldsymbol{\omega}_t^{\mathrm{SHB}}\right\|\right] \leq \sqrt{\left(1+\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}\right)\frac{C_{\mathrm{SHB}}^2}{b}+\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}K_{\mathrm{SHB}}^2}, \tag{9}$$

$$\mathbb{E}\left[\left\|\boldsymbol{\omega}_t^{\mathrm{NSHB}}\right\|\right] \leq \sqrt{\frac{1}{1-\beta}\frac{C_{\mathrm{NSHB}}^2}{b}}. \tag{10}$$

が成り立つ.

SGD に対する議論と同様に，$\boldsymbol{\omega}_t^{\mathrm{SHB}}$ が正規分布に従うという実験的観察 [2] と定理 3.2 から，正規分布に従う確率変数 $\boldsymbol{u}_t \in \mathbb{R}^d$ を使って，式 (9) を満たす $\boldsymbol{\omega}_t^{\mathrm{SHB}}$ は例えば次のように表すことができる.

$$\boldsymbol{\omega}_t^{\mathrm{SHB}} = \sqrt{\left(1+\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}\right)\frac{C_{\mathrm{SHB}}^2}{b}+\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}K_{\mathrm{SHB}}^2}\boldsymbol{u}_t =: \psi^{\mathrm{SHB}}\boldsymbol{u}_t,$$

ただし，$\boldsymbol{u}_t \sim \mathcal{N}\left(\boldsymbol{0}; \frac{1}{\sqrt{d}}I_d\right)$ である．これを式 (8) に代入すると，定義 2.1 から，

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SHB}}}\left[\boldsymbol{y}_{t+1}\right] &= \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SHB}}}\left[\boldsymbol{y}_t\right] - \eta\nabla\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SHB}}}\left[f\left(\boldsymbol{y}_t-\eta\boldsymbol{\omega}_t^{\mathrm{SHB}}\right)\right] \\
&= \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SHB}}}\left[\boldsymbol{y}_t\right] - \eta\nabla\mathbb{E}_{\boldsymbol{u}_t\sim\mathcal{N}\left(\boldsymbol{0};\frac{1}{\sqrt{d}}I_d\right)}\left[f\left(\boldsymbol{y}_t-\eta\psi^{\mathrm{SHB}}\boldsymbol{u}_t\right)\right] \\
&= \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SHB}}}\left[\boldsymbol{y}_t\right] - \eta\nabla\hat{f}_{\eta\psi^{\mathrm{SHB}}}(\boldsymbol{y}_t),
\end{aligned} \tag{11}$$

が成り立つ．したがって，SHB も SGD と同様に，その探索方向ノイズ $\boldsymbol{\omega}_t^{\mathrm{SHB}}$ には目的関数 f を平滑化する効果がある．そして，その度合い δ^{SHB} は $\eta\psi^{\mathrm{SHB}}$，すなわち，学習率 η，バッチサイズ b，確率的勾配の分散 C_{SHB}^2 に加えて，慣性係数 β と勾配ノルムの上界 K_{SHB}^2 で定まる．同様の議論から，NSHB の探索方向ノイズ $\boldsymbol{\omega}_t^{\mathrm{NSHB}}$ による平滑化の度合いも導出することができ，3 つの最適化アルゴリズムの探索方向ノイズによる平滑化の度合いは，次のようになる.

$$\delta^{\mathrm{SGD}} = \eta\sqrt{\frac{C_{\mathrm{SGD}}^2}{b}}, \tag{12}$$

$$\delta^{\mathrm{SHB}} = \eta\sqrt{\left(1+\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}\right)\frac{C_{\mathrm{SHB}}^2}{b}+\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}K_{\mathrm{SHB}}^2}, \tag{13}$$

$$\delta^{\mathrm{NSHB}} = \eta\sqrt{\frac{1}{1-\beta}\frac{C_{\mathrm{NSHB}}^2}{b}}. \tag{14}$$

$\beta = 0$ の δ^{SHB} と δ^{NSHB} は δ^{SGD} に一致することから，SHB と NSHB の結果は SGD の結果の拡張になっている．また，δ^{SHB} に含まれる $\frac{\beta(\beta^2-\beta+1)}{(1-\beta)^2}$ と δ^{NSHB} に含まれる $\frac{1}{1-\beta}$

は，慣性係数 β に対して単調増加するため，SGD with momentum においては，より大きい慣性係数がより大きな平滑化の度合いをもたらすといえる．また，式 (12)-(14) は，学習率 η と慣性係数 β 等のパラメータが平滑化の度合いを介して互いに関係し合っていることを意味している．いくつかの先行研究は，これらのパラメータの間に関係があることを実験的に観察している [37, 38, 19, 39]．例えば，Leclerc と Madry は学習率と慣性係数の間には関係があり，学習率が大きくなればなるほど，慣性係数は小さく設定すると汎化性能が優れることを実験で示した [38, Figure 4]．我々の結果はこの現象に理論的な説明を与えている．すなわち，補題 2.1 から，平滑化の度合いには大きすぎず小さすぎない最適な値があると考えられるため，式 (13) において，学習率 η を大きくとるならば，平滑化の度合い δ^{SHB} が大きくなりすぎないためには慣性係数 β は小さくとる必要があることが分かる．

注意 3.1 式 (12)-(14) において，δ^{SHB} の上界にだけ勾配ノルムの上界 K^2_{SHB} が現れていることを奇妙に思われるかもしれない．δ^{SHB} の上界の K^2_{SHB} を含む項は第 5 章の主張において非常に重要な役割を果たすため，この導出が恣意的ではないことを補足しておこう．δ^{NSHB} の上界に K_{NSHB} 等の余計な項が現れていないのは，$\|\boldsymbol{\omega}_t^{\mathrm{NSHB}}\|^2$ の展開において，NSHB の凸結合の性質を利用できるためである（証明の詳細は文献 [2] の Appendix C.2 章と命題 A.1 を参照されたい）．実際，単に δ^{SHB} の導出に倣って δ^{NSHB} を導出すると，

$$\delta^{\mathrm{NSHB}} = \eta\sqrt{(1+4\beta^2)\,\frac{C^2_{\mathrm{NSHB}}}{b} + K^2_{\mathrm{NSHB}}}$$

となる．したがって，δ^{SHB} の上界に K^2_{SHB} が現れているのは，$\|\boldsymbol{\omega}_t^{\mathrm{SHB}}\|^2$ の展開において凸結合の優れた性質を利用できないためである．SHB と NSHB は実験的に全くの別物であることを示すことができる（図 4, 6 を参照）ため，この問題は我々の理論的な技術不足によるものではなく，むしろ，探索方向ノイズの解析を通じて，アルゴリズムの凸結合の有無が理論的にも実験的にも性能に大きな差をもたらしていることを初めて示すことができた（SHB と NSHB の凸結合については 2.3 節を参照されたい）と考えている．もちろん，K^2_{SHB} を使わずに δ^{SHB} を導出することは重要で興味深い今後の課題である．

さて，我々の興味は，それぞれの平滑化の度合いの大小関係と，平滑化の度合いとモデルの汎化性能の関係である．これを確かめるために，$\delta^{\mathrm{SGD}}, \delta^{\mathrm{SHB}}, \delta^{\mathrm{NSHB}}$ がどのような値をとるか，数値で推定したい．これらを構成するパラメータのうち，学習率 η，バッチサイズ b，慣性係数 β はユーザが任意に設定できる値であるが，確率的勾配の分散 C^2_{opt} と勾配ノルムの上界 K^2_{opt} は未知数である．特に確率的勾配の分散 C^2_{opt} の推定は難しく，

先行研究にも例がない.

第 4 章 確率的勾配の分散の推定

第 3 章では，最適化アルゴリズムごとの平滑化の度合いの式を明らかにした (式 (12)-(14))．本章はそれらの推定のために，理論と実験的観察を交えて確率的勾配の分散 C_{opt}^2 の推定を行う．確率的勾配の分散の推定 (4.3 節) には，クリティカルバッチサイズと呼ばれる，計算量の意味で最適なバッチサイズの推定式 (4.2 節) を利用する．その推定式の導出のためには，3 つの最適化アルゴリズムの収束解析 (4.1 節) が必要となる．

4.1 最適化アルゴリズムの収束解析

本節では, 3 つの最適化アルゴリズムの収束を保証する結果を提供する．SHB と NSHB の解析のために，まずは新たに次の仮定を導入する．

仮定 4.1 (A5) 任意の $\boldsymbol{x} \in \mathbb{R}^d$ と任意の $t \in \mathbb{N}$ に対して，次の式を満たす正の実数 $D(\boldsymbol{x})$ が存在する．

$$\|\boldsymbol{x}_t - \boldsymbol{x}\| \leq D(\boldsymbol{x}).$$

仮定 4.1 は，深層学習における凸最適化と非凸最適化の両方の解析において，アルゴリズムの性能評価のために幅広く使用されてきた一般的な仮定である [40, 41, 42].

次に，SGD，SHB，NSHB の 3 つの最適化アルゴリズムの収束を保証する，以下の 3 つの定理を紹介する．なお，定理 4.1 の証明は文献 [2] の Appendix A.6 章を，定理 4.2 の証明は Appendix A.3 章を，定理 4.3 の証明は Appendix A.5 章を参照されたい.

定理 4.1 仮定 (A1)-(A4) が成り立つと仮定し，点列 $(\boldsymbol{x}_t)_{t\in\mathbb{N}}$ を SGD によって生成された点列とする．このとき，任意の $\boldsymbol{x} \in \mathbb{R}^d$ と任意の $T \geq 1$ に対して，

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\langle \boldsymbol{x}_t - \boldsymbol{x}, \nabla f(\boldsymbol{x}_t)\rangle\right] \leq \frac{\|\boldsymbol{x}_0 - \boldsymbol{x}\|^2}{2\eta T} + \frac{\eta}{2}\left(\frac{C_{\mathrm{SGD}}^2}{b} + K_{\mathrm{SGD}}^2\right).$$

が成り立つ.

定理 4.2 仮定 (A1)-(A5) が成り立つと仮定し，点列 $(\boldsymbol{x}_t)_{t\in\mathbb{N}}$ を SHB によって生成され

た点列とする．このとき，任意の $\boldsymbol{x} \in \mathbb{R}^d$ と任意の $T \geq 1$ に対して，

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\langle \boldsymbol{x}_t - \boldsymbol{x}, \nabla f(\boldsymbol{x}_t)\rangle\right] \leq \frac{\|\boldsymbol{x}_0 - \boldsymbol{x}\|^2}{2\eta T} + \frac{\beta D(\boldsymbol{x})}{1-\beta}\sqrt{\frac{C_{\mathrm{SHB}}^2}{b} + K_{\mathrm{SHB}}^2} + \frac{\eta\left(\beta^2 - \beta + 1\right)}{2\beta(1-\beta)^2}\left(\frac{C_{\mathrm{SHB}}^2}{b} + K_{\mathrm{SHB}}^2\right).$$

が成り立つ．

定理 4.3 仮定 (A1)-(A5) が成り立つと仮定し，点列 $(\boldsymbol{x}_t)_{t\in\mathbb{N}}$ を NSHB によって生成された点列とする．このとき，任意の $\boldsymbol{x} \in \mathbb{R}^d$ と任意の $T \geq 1$ に対して，

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\langle \boldsymbol{x}_t - \boldsymbol{x}, \nabla f(\boldsymbol{x}_t)\rangle\right] \leq \frac{\|\boldsymbol{x}_0 - \boldsymbol{x}\|^2}{2\eta(1-\beta) T} + \frac{\beta D(\boldsymbol{x})}{1-\beta}\sqrt{\frac{C_{\mathrm{NSHB}}^2}{b} + K_{\mathrm{NSHB}}^2} + \frac{\eta}{2(1-\beta)}\left(\frac{C_{\mathrm{NSHB}}^2}{b} + K_{\mathrm{NSHB}}^2\right).$$

が成り立つ．

目的関数 f の最適解 $\boldsymbol{x}^\star \in \mathbb{R}^d$ に対して

$$\nabla f(\boldsymbol{x}^\star) = \boldsymbol{0} \Longleftrightarrow \forall \boldsymbol{x} \in \mathbb{R}^d \colon \langle \nabla f(\boldsymbol{x}^\star), \boldsymbol{x}^\star - \boldsymbol{x}\rangle \leq 0$$

が成り立つ（文献 [2] の命題 A.2 を参照）ことから，評価指標 $\dfrac{1}{T}\displaystyle\sum_{t=0}^{T-1}\mathbb{E}\left[\langle \boldsymbol{x}_t - \boldsymbol{x}, \nabla f(\boldsymbol{x}_t)\rangle\right]$ を小さく抑えることができれば，点列 $(\boldsymbol{x}_t)_{t\in\mathbb{N}}$ は最適解 $\boldsymbol{x}^\star$ に収束するといえる．定理 4.1, 4.2, 4.3 は，反復回数 T とバッチサイズ b を十分大きく，学習率 η を十分小さくとれば，指標の上界を小さくすることができることを意味している．

4.2 クリティカルバッチサイズの推定

本節では，訓練に必要な計算量を最小にするバッチサイズである，クリティカルバッチサイズを推定する方法を述べる．本稿では，クリティカルバッチサイズを各アルゴリズムごとに $b_{\mathrm{opt}}^\star$ で表す．

まず，深層学習における最適化アルゴリズムの計算量である，確率的勾配計算コストの指標である SFO 計算量 (Stochastic First-order Oracle complexity, SFO complexity) を定義する．アルゴリズムが深層学習モデルの訓練にバッチサイズ b を使う場合，アルゴリズムは各時刻で b 個の確率的勾配を計算するため，モデルの訓練に必要な反復回数を T とすると，アルゴリズムの確率的勾配計算コストは Tb となる．これが SFO 計算量であ

る．よって，訓練に必要な計算コストを最小化するためには，SFO 計算量 Tb を最小化すればよい．いくつかの先行研究は，バッチサイズ b を 2 倍にするとモデルの学習に必要な反復回数 T が半分になるが，この現象はクリティカルバッチサイズ $b^\star$ を超えると観測されないことを示した [43, 44, 45]．従って，クリティカルバッチサイズは SFO 計算量を最小化するバッチサイズであり，最適化アルゴリズムは SFO 計算量 Tb の大域的最小解であるクリティカルバッチサイズ $b^\star$ を使うことが望ましいだろう．Zhang らはクリティカルバッチサイズが最適化アルゴリズムに依存することを実験的に示唆し [46]，飯塚はその存在を理論的に証明し [47]，佐藤らは敵対的生成ネットワーク (GAN) の訓練におけるクリティカルバッチサイズ $b^\star$ の下界を，学習率等のパラメータから推定する式を提供した [48]．本稿はこの推定式の導出方法に倣って，一般的な深層学習モデルに対する SGD, SHB，NSHB それぞれのクリティカルバッチサイズの推定式を導出する．

$\epsilon > 0$ とし，定理 4.1, 4.2, 4.3 を用いて，$\dfrac{1}{T_{\mathrm{opt}}} \displaystyle\sum_{t=0}^{T_{\mathrm{opt}}-1} \mathbb{E}\left[\langle \boldsymbol{x}_t - \boldsymbol{x}, \nabla f(\boldsymbol{x}_t)\rangle\right] \leq \epsilon^2$ を満たす T_{opt} を各最適化アルゴリズムによる訓練のために必要な反復回数とする．したがって，ϵ^2 は閾値であり，訓練の停止条件である．このとき，各最適化アルゴリズムのクリティカルバッチサイズ $b^\star_{\mathrm{opt}}$ は $b^\star_{\mathrm{opt}} := \mathrm{argmin}_{b\in[n]} T_{\mathrm{opt}} b$ と定義できる．定理 4.1, 4.2, 4.3 から，クリティカルバッチサイズ $b^\star_{\mathrm{opt}}$ の下界を与える次の定理 4.4 を導出することができる．定理 4.4 の導出のためには複数の補題と定理を説明する必要があるため省略する．定理 4.4 の証明およびその導出については文献 [2] の Appendix B 章を参照されたい．

定理 4.4 仮定 (A1)-(A5) が成り立つと仮定し，SGD，SHB，NSHB による訓練を考える．$\epsilon > 0$ とする．このとき，

$$b^\star_{\mathrm{SGD}} > \frac{\eta C^2_{\mathrm{SGD}}}{\epsilon^2}, \quad b^\star_{\mathrm{SHB}} > \frac{\eta(\beta^2-\beta+1)C^2_{\mathrm{SHB}}}{\beta(1-\beta)^2\epsilon^2}, \quad b^\star_{\mathrm{NSHB}} > \frac{\eta C^2_{\mathrm{NSHB}}}{(1-\beta)\epsilon^2}.$$

が成り立つ．

定理 4.4 は，例えば SHB のクリティカルバッチサイズ $b^\star_{\mathrm{SHB}}$ は学習率 η，慣性係数 β，確率的勾配の分散 C^2_{SHB}，そして閾値 ϵ によって定まることを示している．クリティカルバッチサイズと閾値 ϵ の間に関係があり，より厳しい停止条件，すなわち小さい ϵ がクリティカルバッチサイズの増加をもたらすことは先行研究によって実験的に観察されていた [46]．定理 4.4 によると，ϵ が小さいときクリティカルバッチサイズの下界は大きくなるため，我々の結果はこの実験的観察の理論的な裏付けになっている．クリティカルバッチサイズは SFO 計算量を最小にするバッチサイズであるから，実務家にとって，それを学習率等のパラメータから推定できるこの定理は非常に価値のあるものである．ところが，

この推定式は未知数である確率的勾配の分散 C_{opt}^2 も含んでいるため，訓練前にクリティカルバッチサイズを推定することはできないという限界がある．

しかし，クリティカルバッチサイズ $b_{\mathrm{opt}}^\star$ は数値実験で測定できるため，定理 4.4 の推定式において，未知数は確率的勾配の分散 C_{opt}^2 のみとなる．つまり，定理 4.4 はクリティカルバッチサイズを事前に知るのには不十分だが，確率的勾配の分散 C_{opt}^2 の推定式としては利用価値がある．

4.3 確率的勾配の分散の推定

本節では，定理 4.4 を利用して確率的勾配の分散を推定する．そのために，まずはアルゴリズムごとのクリティカルバッチサイズを数値実験で測定する．

訓練データには，100 クラス，計 60,000 枚の 32×32 ピクセルのカラー画像からなる CIFAR100 データセット [49] を使用する．深層学習モデルには，一般的な CNN ベースのモデルで，約 1170 万個のパラメータもつ ResNet18[50] を使用する．全ての最適化アルゴリズムで共通して学習率は $\eta = 0.1$ とし，SHB と NSHB の慣性係数はどちらも $\beta = 0.9$ に設定した．異なるバッチサイズ $b \in \{2^3, \ldots, 2^{13}\}$ と 3 つの最適化アルゴリズムに対して，時刻 t における過去 t 回分の確率的勾配のノルムの平均が $\epsilon = 0.5$ 以下となるまでに必要な反復回数 T_{opt} を測定し，SFO 計算量 $T_{\mathrm{opt}}b$ を計算する．そして，SFO 計算量を最小にするバッチサイズが，クリティカルバッチサイズ $b_{\mathrm{opt}}^\star$ である．図 4 は，SFO 計算量 $T_{\mathrm{opt}}b$ とバッチサイズ b の関係をプロットしたものである．

図 4 から，SGD，SHB，NSHB のクリティカルバッチサイズはそれぞれ $b_{\mathrm{SGD}}^\star = 2^9, b_{\mathrm{SHB}}^\star = 2^{10}, b_{\mathrm{NSHB}}^\star = 2^9$ であることがわかる．この値を，訓練に使われた学習率等のパラメータとともに定理 4.4 に代入すると，アルゴリズムごとの確率的勾配の分散の上界は次のように推定することができる．

$$
\begin{aligned}
C_{\mathrm{SGD}}^2 &< \frac{b_{\mathrm{SGD}}^\star \epsilon^2}{\eta} = \frac{2^9 \cdot (0.5)^2}{0.1} = 1280, \\
C_{\mathrm{SHB}}^2 &< \frac{b_{\mathrm{SHB}}^\star \epsilon^2 \beta (1-\beta)^2}{\eta(\beta^2 - \beta + 1)} = \frac{2^{10} \cdot (0.5)^2 \cdot 0.9 \cdot (0.1)^2}{0.1 \cdot 0.91} = 25.318, \\
C_{\mathrm{NSHB}}^2 &< \frac{b_{\mathrm{NSHB}}^\star \epsilon^2 (1-\beta)}{\eta} = \frac{2^9 \cdot (0.5)^2 \cdot (1-0.9)}{0.1} = 128.
\end{aligned}
$$

ここで，これらの値は任意のデータセットと任意の訓練データに対して有効なものではなく，問題依存の値であることに注意する．すなわち，上式で得られたそれぞれの確率的勾配の分散は，あくまでも CIFAR100 データセットで ResNet18 を訓練するときの推定値である．

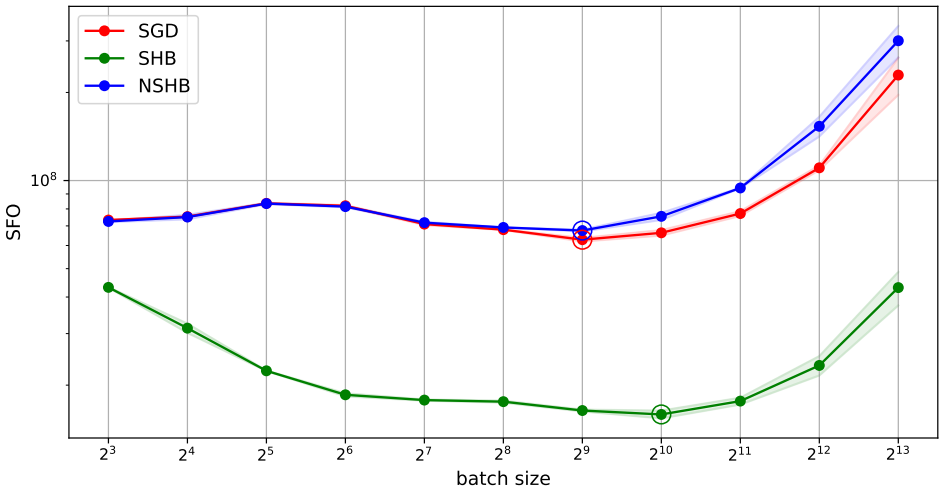

図 4　CIFAR100 データセットによる ResNet18 の訓練において, SGD, SHB, NSHB をそれぞれ使うときの SFO 計算量と，バッチサイズの関係．図中の二重丸は SFO 計算量の最小値を表している．実線は 3 回の実行の平均値を表しており，3 回の実行の最大値と最小値の間を薄く塗りつぶしている．また，縦軸は底が 10 の対数グラフ，横軸は底が 2 の対数グラフであることに注意する．

第 5 章　探索方向ノイズと汎化性能

第 4 章では，推定が困難とされている確率的勾配の分散 C_{opt}^2 をアルゴリズムごとに推定することができた．本章では，それらを第 3 章で明らかになった平滑化の度合いの式 (12)-(14) に代入することで，アルゴリズムごとの平滑化の度合いを数値で推定し，モデルの汎化性能との関係を明らかにしていく．また，それによって，実験的には観察されていたが理論的には解明されていなかったいくつかの現象を，理論的に説明できることを示す．

5.1　平滑化の度合いと汎化性能

図 5 は，推定した確率的勾配の分散を代入することで導出した平滑化の度合いとバッチサイズの関係をプロットしたものである．ただし，全ての最適化アルゴリズムで学習率は $\eta = 0.1$，SHB と NSHB の慣性係数は $\beta = 0.1$ としている．特に，バッチサイズが大きくなるにつれて，δ^{SGD} と δ^{NSHB} は 0 に近づいていくのに対して，δ^{SHB} はある一定の値よりも小さくならないことに注目してほしい．これは．δ^{SHB} だけがバッチサイズ b と独立な項を有するためである (式 13 を参照)．また，δ^{SGD} と δ^{NSHB} が完全に一致していること，δ^{SHB} は全てのバッチサイズで大きい値をとっていることにも注目したい．

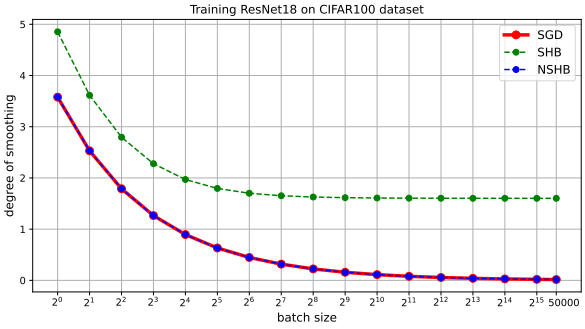

図 5　CIFAR100 データセットによる ResNet18 の訓練における，学習率 $\eta = 0.1$, 慣性係数 $\beta = 0.9$ のときの，それぞれの平滑化の度合い $\delta^{\mathrm{SGD}}, \delta^{\mathrm{SHB}}, \delta^{\mathrm{NSHB}}$ とバッチサイズの関係.

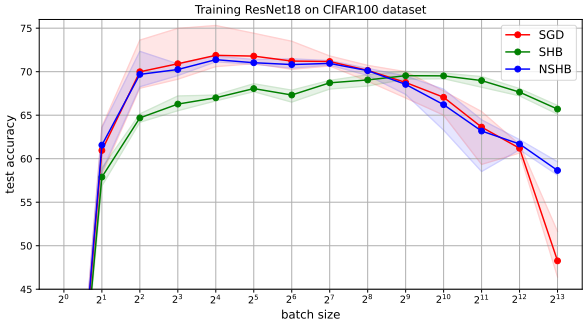

図 6　SGD, SHB, NSHB それぞれによって，CIFAR100 データセットで ResNet18 を 200 エポック訓練して得られるテスト精度とバッチサイズの関係．実線は 7 回の実行の平均値を表しており，7 回の実行の最大値と最小値の間を薄く塗りつぶしている．

これらを踏まえて，図 6 にプロットしたテスト精度とバッチサイズの関係を見てみよう．特に注目したいのは，SGD と NSHB のテスト精度が，バッチサイズが大きくなるにつれて悪化しているのに対して，SHB のテスト精度はさほどバッチサイズの影響を受けていないように見えることである．この現象はいくつかの先行研究でも観察されている [43, 51, 19]．これは，バッチサイズが大きいときでも δ^{SHB} だけは十分に大きく保たれており，目的関数が十分に平滑化されていることが原因であろう．逆に，バッチサイズが大きいとき，δ^{SGD} と δ^{NSHB} は小さすぎるため，目的関数の平滑化が不十分になり，アルゴリズムによる点列はその近傍が急峻な局所的最適解に陥り，結果としてテスト精度が優れない．また，SGD と NSHB のテスト精度はほぼ一致しているが，これは δ^{SGD} と δ^{NSHB} が完全に一致しているためだろう．最後に，SHB のテスト精度が，全てのバッチサイズに対して 70% を超えることがなく，特にバッチサイズが小さいとき，SGD と NSHB の方が SHB よりもテスト精度が高いことに注目したい．これは，δ^{SHB} が δ^{SGD} と δ^{NSHB} よりも常に大きく，特にバッチサイズが小さいとき，δ^{SHB} が大きすぎることが原因であると考えられる．十分な平滑化のためには十分な大きさの平滑化の度合いが必要だが，大きければ大きいほど良いわけではないことが，補題 2.1 から分かる．すなわち，大きすぎる平滑化の度合いは，元の関数との大きすぎる乖離をもたらし，結果として適切に最適化がなされないために，テスト精度が優れないと考察できる．したがって，大きすぎず小さすぎない平滑化の度合いが，高い汎化性能をもたらす．

図 7 は，図 5，6 にテスト精度と平滑化の度合いのグラフを加えたものである．ここで，

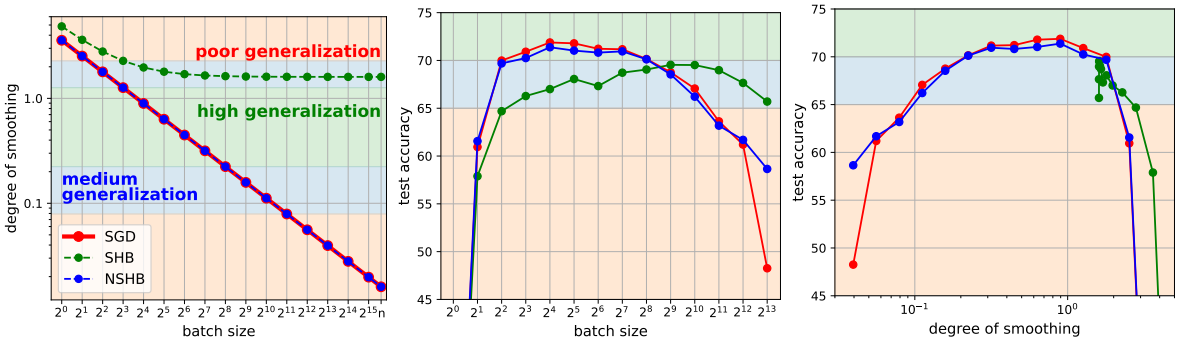

図 7　**左:** 3 つの最適化アルゴリズムの探索方向ノイズによってもたらされる平滑化の度合いとバッチサイズの関係．**中央:** 3 つの最適化アルゴリズムによるテスト精度とバッチサイズの関係．**右:** 3 つの最適化アルゴリズムによるテスト精度と平滑化の度合いの関係．テスト精度 70% 以上を高い汎化性能（緑），テスト精度 65% 以上を中間の汎化性能（青），テスト精度 65% 未満を悪い汎化性能（赤）と定義して背景を塗りつぶしている．

図 7 は，分かりやすさのために底が 10 の対数グラフになっていることに注意されたい．図 7 の**左**と**右**のグラフから，CIFAR100 データセットによる ResNet18 の訓練においては，およそ 0.2 から 1 程度の平滑化の度合いが最適であることが分かる．

以上の考察から分かる通り，深層学習モデルの汎化性能は，最適化アルゴリズムの探索方向ノイズによる平滑化の度合いで説明できるといえよう．これは言わば，深層学習モデルの訓練における隠された法則である．なお，他のデータセットと他の深層学習モデルに対する数値実験でも，同様の結果が得られている．結果の詳細は文献 [2] を参照されたい．

5.2 SGD with momentum の確率的ノイズと汎化性能の矛盾の解決

1.1.5 節で述べたように，SGD with momentum は確率的ノイズと汎化性能に関する矛盾を抱えていた．本節では，本稿のこれまでの結果によってその矛盾を解消する．

第 3 章の定理 3.1 から，最適化アルゴリズムの勾配ノイズは，アルゴリズムごとの確率的勾配の分散 C_{opt}^2 であることが分かっていた．そして，第 4 章では，CIFAR100 データセットで ResNet18 を訓練するときの確率的勾配の分散は，アルゴリズムごとにそれぞれ $C_{\mathrm{SGD}}^2 < 1280, C_{\mathrm{SHB}}^2 < 25.318, C_{\mathrm{NSHB}}^2 < 128$ であることが分かっていた．したがって，確かに SHB と NSHB の勾配ノイズは SGD のそれよりも小さくなっており，慣性項を加えることで勾配ノイズが削減されているという一般的な主張は正しいことが分かる．

一方，第 3 章の定理 3.2 と式 (12)-(14) から，最適化アルゴリズムの探索方向ノイズは目的関数の平滑化に寄与しており，その度合いはまさに探索方向ノイズの大きさによって

定まっていることが分かる．そして，5.1 節の図 5 から，SHB の平滑化の度合い δ^{SHB} は SGD の平滑化の度合い δ^{SGD} よりも常に大きくなっている．また，同節の図 7 から，平滑化の度合いがモデルの汎化性能に寄与していることが分かる．したがって，慣性項を加えると勾配ノイズは削減されるが，汎化性能に寄与する探索方向ノイズは増大する．

以上のことから，「慣性項を加えると確率的ノイズが削減される」という主張と，「確率的ノイズが高い汎化性能をもたらす」という主張は競合せず，矛盾は解消された．すなわち，慣性項を加えることで削減される確率的ノイズと，汎化性能に寄与する確率的ノイズは別物であることを示すことができた．なお，NSHB の汎化性能は SGD とほぼ変わらないため，NSHB に対しては上述の矛盾は発生していなかったことが分かった．

5.3 Sharpness と平滑化の度合い

1.1.3 節で述べたように，経験損失関数の局所的最適解の近傍の形状は，モデルの汎化性能と密接な関係にあると考えられている．近似解の近傍での関数の滑らかさの指標 Sharpness には，いくつか種類があるが，本稿は Kwon らが提案した “adaptive sharpness”[14] に焦点を当てて議論を進める．adaptive sharpness は既存の複数の Sharpness の拡張になっており，モデルの汎化性能と特に強い相関があることが知られている．訓練データ集合を $\mathcal{S}$ とすると，モデルの任意のパラメータ $\boldsymbol{w} \in \mathbb{R}^d$ に対して，半径 $\rho \in \mathbb{R}$ とあるベクトル $\boldsymbol{c} \in \mathbb{R}^d$ をもつ adaptive sharpness は,

$$S^{\rho}_{\max}(\boldsymbol{w}, \boldsymbol{c}) := \mathbb{E}_{\mathcal{S}}\left[\max_{\|\boldsymbol{\delta} \odot \boldsymbol{c}^{-1}\|_p \leq \rho} f(\boldsymbol{w}+\boldsymbol{\delta}) - f(\boldsymbol{w})\right]$$

で定義される．ただし，$\odot/^{-1}$ はそれぞれベクトルの要素ごとの積/逆数を意味する．つまり，あるパラメータ $\boldsymbol{w}$ における関数値 $f(\boldsymbol{w})$ と，$\boldsymbol{w}$ の周囲，半径 ρ の近傍での関数値 $f(\boldsymbol{w}+\boldsymbol{\delta})$ の差の最大値が adaptive sharpness である．よって，より大きい adaptive sharpness は，モデルのパラメータ $\boldsymbol{w}$ の半径 ρ の近傍で，関数がより鋭いことを意味する．いくつかの先行研究は，より小さい Sharpness が高い汎化性能をもたらすことを実験的に観察している [10, 15, 16]．すなわち，これらの実験的観察は，「その近傍が平坦な最適解は，その近傍が急峻な最適解よりも優れた汎化性をもたらす」という仮説を支持するものである．

では，我々が導出した，最適化アルゴリズムの探索方向ノイズによる平滑化の度合いと，Sharpness の間にはどのような関係があるだろうか．これを確かめるために，学習率 $\eta \in \{0.01, 0.05, 0.1, 0.5\}$ とバッチサイズ $b \in \{2^1, \ldots, 2^{13}\}$ の組み合わせで，SGD によって ResNet18 を CIFAR100 データセットで 200 エポック訓練し，SGD が到達した近似

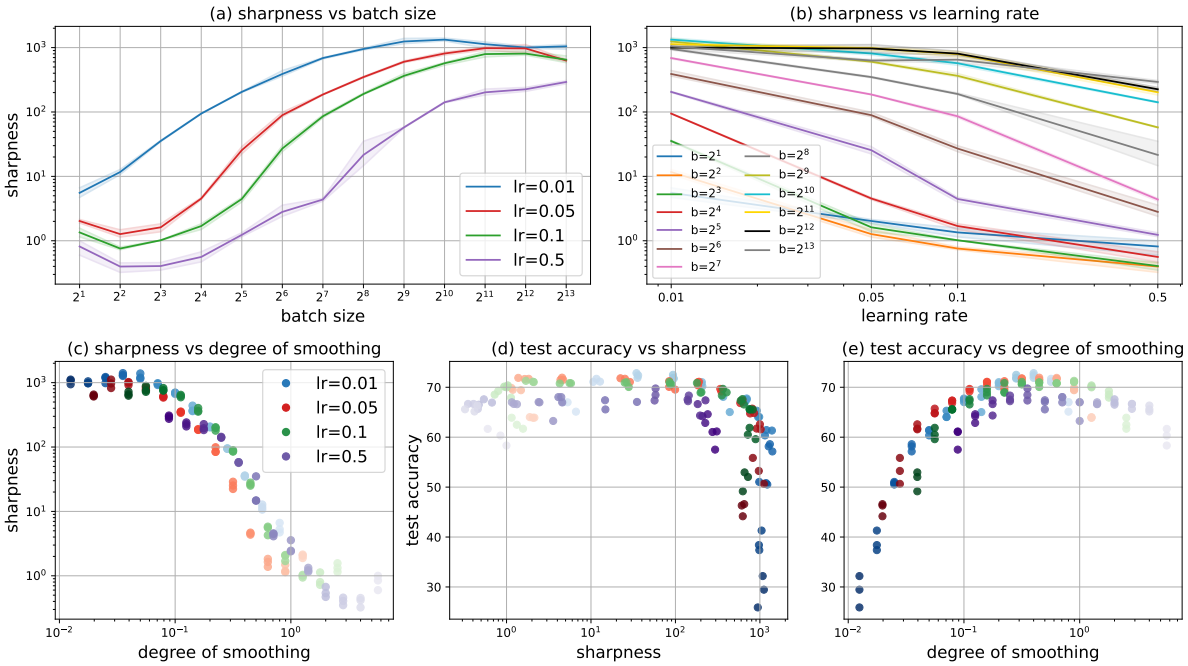

図 8　**(a)** CIFAR100 データセットによる ResNet18 の 200 エポックの訓練で得られた近似解の近傍で計測された Sharpness と，訓練に使われたバッチサイズの関係．**(b)** Sharpness と，訓練に使われた学習率の関係．**(c)** Sharpness と，学習率，バッチサイズ，推定された確率的勾配の分散から計算された平滑化の度合いの関係．**(d)** テスト精度と Sharpness の関係．**(e)** テスト精度と平滑化の度合いの関係．実線は 7 回の実行の平均値を表しており，7 回の実行の最大値と最小値の間を薄く塗りつぶしている．散布図における色の濃淡はバッチサイズの大小を表しており，色が濃いほどバッチサイズが大きいことを意味する．全てのグラフは同じ実験結果を異なる視点から示している．

解の周り，半径 $\rho = 0.0002$ の近傍で adaptive sharpness を計測した．また，4 つの学習率に対して，第 4 章の手順で確率的勾配の分散を推定し，平滑化の度合いを数値で導出した．その結果を図 8 にまとめる．なお，図中の “lr” は学習率を意味している．

図 8(a) は，バッチサイズが大きいほど Sharpness が大きくなること，(b) は学習率が大きいほど Sharpness が小さくなること，(c) は Sharpness が小さいほど平滑化の度合いが大きくなることを示している．これらの結果は，平滑化の度合い δ^{SGD} が学習率 η に比例し，バッチサイズ b に反比例するという我々の理論結果を保証し，$\delta^{\mathrm{SGD}} := \eta\sqrt{C_{\mathrm{SGD}}^2/b}$ が関数の平滑化の度合いを決定しているという我々の理論を補強している．また，Keskar らは，大きすぎるバッチサイズが Shapness の増加をもたらし，汎化性能を悪化させることを実験的に観察している [10]．図 8(a) はこの実験的観察と一致し，この現象も我々の理論によって理論的な説明を与えることができる．すなわち，大きすぎるバッチサイズが

Sharpness の増加をもたらすのは，バッチサイズが平滑化の度合いに対して反比例するためであるといえる．図 8(d) は，モデルの汎化性能と近似解付近の Sharpness との間に明確な相関がないことを示している．この結果は，ResNet 等の現代的な深層学習モデルの訓練においては，Sharpness と汎化性能の間には相関が見られないことを示した先行研究 [17] と一致する．一方，図 8(e) は，図 7 **右**と同様に，汎化性能と平滑化の度合いとの間に優れた相関がある（汎化性能は平滑化の度合いに対して凹関数になっている）ことを示しており，大きすぎず小さすぎない平滑化の度合いが高い汎化性能をもたらすことを示している．5.1 節で述べた通り，この実験的観察には理論的な裏付けが可能である (補題 2.1 を参照)．

Andriushchenko らは，Sharpness と汎化性能に関する既存の先行研究がいずれも小規模な深層学習モデルを対象にしていることに注目し，現代的な大規模な深層学習モデルを用いた広範な数値実験を行い，それらに対しては Sharpness と汎化性能の間にはほぼ相関が見られないことを発見し，Sharpness は汎化性能の良い指標ではない可能性を示唆した [17]．またそれによって，「より平坦な最適解がより良い汎化性能をもたらす」という仮説の正しさに疑問符を投げかけている．図 8(c) から，平滑化の度合いと Sharpness の両方が目的関数の滑らかさ/鋭さを表していることは明らかである．さらに，図 8(d) と (e) から，上述の先行研究と同様に，Sharpness と汎化性能の間には相関がない一方で，平滑化の度合いと汎化性能の間には相関があることが分かる．したがって，平滑化の度合いは汎化性能を理論的に評価するためのより良い指標であるといえ，またそのため，「より平坦な最適解がより良い汎化性能をもたらす」という仮説を破棄するのは早計かもしれない．

第 6 章 まとめ

本稿は，SGD，SHB，NSHB の 3 つの最適化アルゴリズムに焦点を当て，これらが暗黙のうちに抱えている探索方向ノイズが，目的関数を平滑化しているとみなせること，その平滑化の度合いは学習率やバッチサイズ等のパラメータで定まることを示した．また，アルゴリズムの収束解析と，クリティカルバッチサイズの推定を介して，推定が困難な確率的勾配の分散の推定を行い，それによってアルゴリズムごとの平滑化の度合いを数値で導出することに成功した．その結果，深層学習モデルの汎化性能は平滑化の度合いと密接な関係にあり，大きすぎず小さすぎない適度な平滑化の度合いが高い汎化性能をもたらすことを示した．さらに本稿の結果によって，SGD with momentum の確率的ノイズと汎化性能の関する矛盾などの，これまで実験的には観察されていたが理論的には解明されていなかったいくつかの現象に理論的な説明を与えることができた．

今後は，最も高い汎化性能をもたらす最適な平滑化の度合いを事前に推定するための研究が必要であろう．本稿で述べた通り，平滑化の度合いは主に学習率やバッチサイズ等のユーザーが任意に設定できるパラメータで構成されている．したがって，モデルを訓練する際に，グリッドサーチなどによって学習率やバッチサイズ等のパラメータの最適な組み合わせを探索することは，最適な平滑化の度合いを探索することであるともいえる．よって，最適な平滑化の度合いの理解が進めば，パラメータの探索にかかる膨大な計算コストを削減できる可能性がある．また同様の理由から，Sharpness と比較して平滑化の度合いはモデルの汎化性能の新しい指標として有用であることを強調したい．

参考文献

[1] Naoki Sato and Hideaki Iiduka. Using stochastic gradient descent to smooth nonconvex functions: Analysis of implicit graduated optimization with optimal noise scheduling. `https://arxiv.org/abs/2311.08745`, 2023.

[2] Naoki Sato and Hideaki Iiduka. Role of momentum in smoothing objective function and generalizability of deep neural networks. `https://arxiv.org/abs/2402.02325`, 2024.

[3] 佐藤尚樹 and 飯塚秀明. **確率的勾配降下法の平滑化効果を利用した段階的最適化手法によるディープニューラルネットワークの大域的最適化**. 日本オペレーションズ・リサーチ学会 2024 年春季研究発表会, 筑波大学 筑波キャンパス 春日エリア, 2024.

[4] 佐藤尚樹 and 飯塚秀明. **目的関数の平滑化とディープニューラルネットワークの汎化性能におけるモーメンタム法の慣性項の役割**. 日本オペレーションズ・リサーチ学会 2024 年秋季研究発表会, 南山大学, 2024.

[5] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22:400–407, 1951.

[6] B.T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[7] A.M. Gupal and L. T. Bazhenov. A stochastic analog of the conjugate gradient method. *Cybernetics*, 8(1):138–140, 1972.

[8] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(533-536), 1986.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *MIT Press*, 9(1):1–42, 1997.

[10] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[11] Tengyuan Liang, Tomaso A. Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, volume 89, pages 888–896, 2019.

[12] Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Normalized flat minima: Exploring scale invariant definition of flat minima for neural networks using pac-bayesian analysis. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 9636–9647, 2020.

[13] Henning Petzka, Michael Kamp, Linara Adilova, Cristian Sminchisescu, and Mario Boley. Relative flatness and generalization. In *Proceedings of the 34th Conference on Neural Information Processing Systems*, pages 18420–18432, 2021.

[14] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. ASAM: adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 5905–5914, 2021.

[15] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *Proceedings of the 34th Conference on Uncertainly in Artificial Intelligence*, pages 876–885, 2018.

[16] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, pages 6391–6401, 2018.

[17] Maksym Andriushchenko, Francesco Croce, Maximilian Müller, Matthias Hein, and Nicolas Flammarion. A modern look at the relationship between sharpness and generalization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 840–902, 2023.

[18] Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank J. Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? In *Proceedings of the 33rd Annual Conference on*

Neural Information Processing Systems, 2020.

[19] Frederik Kunstner, Jacques Chen, Jonathan Wilder Lavington, and Mark Schmidt. Noise is not the main factor behind the gap between SGD and adam on transformers, but sign descent might be. In *Proceedings of the 8th International Conference on Learning Representations*, 2023.

[20] Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 1225–1234, 2016.

[21] Wenlong Mou, Liwei Wang, Xiyu Zhai, and Kai Zheng. Generalization bounds of SGLD for non-convex learning: Two theoretical viewpoints. In *Proceedings of the 31st Annual Conference on Learning Theory*, volume 75, pages 605–638, 2018.

[22] Robert Kleinberg, Yuanzhi Li, and Yang Yuan. An alternative view: When does SGD escape local minima? In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 2703–2712, 2018.

[23] Zhijun Wu. The effective energy transformation scheme as a special continuation approach to global optimization with application to molecular conformation. *SIAM Journal on Optimization*, 6(3):748–768, 1996.

[24] Aaron Defazio. Momentum via primal averaging: Theoretical insights and learning rate schedules for non-convex optimization. `https://arxiv.org/abs/2010.00406`, 2020.

[25] Ashok Cutkosky and Harsh Mehta. Momentum improves normalized SGD. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 2260–2268, 2020.

[26] Y. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. *Doklady AN USSR*, 269:543–547, 1983.

[27] Yurii E. Nesterov. *Introductory Lectures on Convex Optimization - A Basic Course*, volume 87 of *Applied Optimization*. Springer, 2004.

[28] Yurii E. Nesterov. Gradient methods for minimizing composite functions. *Mathmatical Programming*, 140(1):125–161, 2013.

[29] Ilya Sutskever, James Martens, George E. Dahl, and Geoffrey E. Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pages

1139–1147, 2013.

[30] Saman Cyrus, Bin Hu, Bryan Van Scoy, and Laurent Lessard. A robust accelerated optimization algorithm for strongly convex functions. In *Annual American Control Conference*, pages 1376–1381, 2018.

[31] Igor Gitman, Hunter Lang, Pengchuan Zhang, and Lin Xiao. Understanding the role of momentum in stochastic gradient methods. In *Advances in Neural Information Processing Systems*, volume 32, pages 9630–9640, 2019.

[32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8024–8035, 2019.

[33] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation*, pages 265–283, 2016.

[34] Harshvardhan and Sebastian U. Stich. Escaping local minima with stochastic noise. In *the 13th International OPT Workshop on Optimization for Machine Learning in NeurIPS 2021*, 2021.

[35] Aimo A. Törn and Antanas Zilinskas. *Global Optimization*, volume 350 of *Lecture Notes in Computer Science*. Springer, 1989.

[36] Günter Rudolph. *Globale optimierung mit parallelen evolutionsstrategien*. PhD thesis, Universität Dortmund Fachbereich Informatik, 7 1990.

[37] Rahul Kidambi, Praneeth Netrapalli, Prateek Jain, and Sham M. Kakade. On the insufficiency of existing momentum schemes for stochastic optimization. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[38] Guillaume Leclerc and Aleksander Madry. The two regimes of deep network

training. https://arxiv.org/abs/2002.10376, 2020.

[39] Jingwen Fu, Bohan Wang, Huishuai Zhang, Zhizheng Zhang, Wei Chen, and Nanning Zheng. When and why momentum accelerates SGD: An empirical study. https://arxiv.org/abs/2306.09000, 2023.

[40] Diederik P Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, pages 1–15, 2015.

[41] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[42] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar Tatikonda, Nicha C. Dvornek, Xenophon Papademetris, and James S. Duncan. AdaBelief optimizer: Adapting stepsizes by the belief in observed gradients. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[43] Christopher J. Shallue, Jaehoon Lee, Joseph M. Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E. Dahl. Measuring the effects of data parallelism on neural network training. *Journal of Machine Learning Research*, 20(112):1–49, 2019.

[44] Siyuan Ma, Raef Bassily, and Mikhail Belkin. The power of interpolation: Understanding the effectiveness of SGD in modern over-parametrized learning. *Proceedings of the 35th International Conference on Machine Learning*, 80:3331–3340, 2018.

[45] Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. http://arxiv.org/abs/1812.06162, 2018.

[46] Guodong Zhang, Lala Li, Zachary Nado, James Martens, Sushant Sachdeva, George E. Dahl, Christopher J. Shallue, and Roger B. Grosse. Which algorithmic choices matter at which batch sizes? insights from a noisy quadratic model. In *Advances in Neural Information Processing Systems*, volume 32, pages 8194–8205, 2019.

[47] Hideaki Iiduka. Critical bach size minimizes stochastic first-order oracle complexity of deep learning optimizer using hyperparameters close to one. https://arxiv.org/abs/2208.09814, 2022.

[48] Naoki Sato and Hideaki Iiduka. Existence and estimation of critical batch size for training generative adversarial networks with two time-scale update rule. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 30080–30104, 2023.

[49] Alex Krizhevsky. Learning multiple layers of features from tiny images. `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`, 2009.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[51] Samy Jelassi and Yuanzhi Li. Towards understanding how momentum improves generalization in deep learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 9965–10040, 2022.

[52] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczynski. *Lectures on Stochastic Programming - Modeling and Theory*. MOS-SIAM Series on Optimization. SIAM, 2009.

付録 A 式 (2) の導出

現在のパラメータを $\boldsymbol{x}_t$ とする．GD で点列を更新した先を $\boldsymbol{y}_t$，SGD で点列を更新した先を $\boldsymbol{x}_{t+1}$ とする，すなわち，

$$\begin{aligned}\boldsymbol{y}_t &:= \boldsymbol{x}_t - \eta\nabla f(\boldsymbol{x}_t),\\ \boldsymbol{x}_{t+1} &:= \boldsymbol{x}_t - \eta\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)\\ &= \boldsymbol{x}_t - \eta(\nabla f(\boldsymbol{x}_t) + \boldsymbol{\omega}_t^{\mathrm{SGD}})\end{aligned}$$

とする．すると， $\boldsymbol{\omega}_t^{\mathrm{SGD}} := \nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t) - \nabla f(\boldsymbol{x}_t)$ から，

$$\begin{aligned}\boldsymbol{x}_{t+1} &:= \boldsymbol{x}_t - \eta\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)\\ &= (\boldsymbol{y}_t + \eta\nabla f(\boldsymbol{x}_t)) - \eta\nabla f_{\mathcal{S}_t}(\boldsymbol{x}_t)\\ &= \boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SGD}}\end{aligned} \tag{15}$$

が成り立つ．したがって，

$$\begin{aligned}\boldsymbol{y}_{t+1} &= \boldsymbol{x}_{t+1} - \eta\nabla f(\boldsymbol{x}_{t+1})\\ &= \boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SGD}} - \eta\nabla f(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SGD}})\end{aligned}$$

が成り立つ．両辺に $\boldsymbol{\omega}_t^{\mathrm{SGD}}$ に関する期待値を取ると，$\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}[\boldsymbol{\omega}_t^{\mathrm{SGD}}] = \mathbf{0}$ から，

$$\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[\boldsymbol{y}_{t+1}\right] = \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[\boldsymbol{y}_t\right] - \eta\nabla\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[f(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SGD}})\right]$$

が成り立つ．ここで，$\mathbb{E}_{\boldsymbol{\omega}_t}\left[\nabla f(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t)\right] = \nabla\mathbb{E}_{\boldsymbol{\omega}_t}\left[f(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t)\right]$ を利用していることに注意する．この等式は関数 f がリプシッツ連続かつ微分可能であるときに成り立つ [52, Theorem 7.49]．そして本稿における仮定 (A1) より，これらの条件は満たされている．式 (2) の導出は以上である．

また，式 (15) と $\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[\boldsymbol{\omega}_t^{\mathrm{SGD}}\right] = \mathbf{0}$ から，

$$\mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[\boldsymbol{x}_{t+1}\right] = \boldsymbol{y}_t.$$

が成り立つ．したがって，SGD によって得られる関数 f のパラメータ $\boldsymbol{x}_{t+1}$ と，GD によって得られる関数 $\hat{f}(\boldsymbol{y}_t) := \mathbb{E}_{\boldsymbol{\omega}_t^{\mathrm{SGD}}}\left[f(\boldsymbol{y}_t - \eta\boldsymbol{\omega}_t^{\mathrm{SGD}})\right]$ のパラメータ $\boldsymbol{y}_t$ は期待値の意味では等しいといえる．