

2015年度

博士学位論文

実数と複素数の多層パーセプトロンの
特異領域を利用した探索法の研究

中部大学大学院

工学研究科 情報工学専攻

佐藤 聖也

論文要旨

重みと入出力が全て実数の多層パーセプトロン (MLP: Multi-Layer Perceptron) は、隠れユニットが十分あれば任意の非線形の回帰や分類問題を解けるため、関数近似やデータマイニング、画像認識などに幅広く応用されている。しかし、実 MLP の重みが形成する探索空間には、勾配がゼロの連続領域である特異領域が多数存在し、学習が停滞してしまう問題がある。そのため、BP(back propagation) よりも強力な、準 Newton 法の一種の BPQ(BP based on Quasi-Newton) を用いても、常に良質の解が得られるとは限らない。

本稿では、まず、実 MLP を対象に、学習が停滞する特異領域を逆に利用する探索法である特異階段追跡法 (SSF: singularity stairs following) を提案する。隠れユニットが J 個の実 MLP(実 MLP(J)) の特異領域は実 MLP($J-1$) の最適解に可約性写像を適用すると形成される。特異領域上のほとんどの点は降下ルートが存在する鞍点であり、特異領域上の実 MLP(J) の入出力は実 MLP($J-1$) の最適解と等価である。この性質を利用して特異領域から探索空間を降下すれば、SSF は隠れユニットの増加に伴って訓練誤差を単調減少させられる。初期に作った SSF1.2 を用いた計算機実験では、BP や BPQ よりも安定して良質の解が得られた。しかし、SSF1.2 には隠れユニットの増加に伴って処理時間が大幅に増加する傾向があったため、探索枝刈り手法を導入した SSF1.3 を提案した。また、特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し、探索数に上限を設定する手法により一層高速化した SSF1.4 を提案した。計算機実験では、SSF1.4 により、従来法よりも 2~5 倍高速に、かつ安定して良質の解が得られた。

次に、重みと入出力が全て複素数である複素 MLP は実 MLP にはない周期性学習などの特長を有するため、様々な応用が期待される。しかし、実 MLP と同様、勾配がゼロの特異領域や局所最適解が多数存在するため、常に良質の解が得られるとは限らない。そこで、複素 MLP の探索法として、SSF1.2, SSF1.3, SSF1.4 をそれぞれ複素 MLP に拡張した C-SSF1.0, C-SSF1.1, C-SSF1.3 を提案した。計算機実験では、C-SSF1.3 を用いれば、従来法よりも常に高速かつ安定して良質の解が得られた。

最後に、決定論的カオス挙動を示すローレンツ方程式と 2 重振り子から生成される時系列データを用い、SSF1.4 と C-SSF1.3 を評価する計算機実験を行った。従来法では学習性能に限界があり、良質の解が得られず、予測性能も劣悪であったが、SSF1.4 と C-SSF1.3 では訓練誤差が単調減少し、ローレンツ方程式から生成される時系列を共に精度良く予測できた。2 重振り子の実験では、1 ステップ先の予測は共に良い精度であったが、フリーラン予測では C-SSF1.3 のみが良好であった。

目次

第1章 序論	1
1.1 研究背景	1
1.2 研究目的	2
1.3 論文構成	3
第2章 実多層パーセプトロン (実 MLP)	5
2.1 実 MLP の出力	6
2.2 実 MLP の学習法	6
2.2.1 目的関数	6
2.2.2 BP	6
2.2.3 Newton 法	7
2.2.4 準 Newton 法	7
2.2.5 直線探索	8
2.3 実 MLP の問題点	9
2.3.1 条件数	9
2.3.2 局所最適解	9
2.3.3 特異領域	10
第3章 複素多層パーセプトロン (複素 MLP)	16
3.1 複素 MLP の出力	16
3.1.1 複素 MLP の活性化関数	16
3.2 複素 MLP の目的関数	18
3.3 Wirtinger 微分による勾配と Hesse 行列の計算	18
3.4 複素 MLP の学習法	20

3.4.1	C-BP	20
3.4.2	C-BFGS	20
3.5	複素 MLP の特異領域	20
第 4 章	特異階段追跡法 (SSF)	22
4.1	SSF1.2	22
4.1.1	特異領域からの探索	24
4.1.2	SSF1.2 の処理の概要	25
4.1.3	計算機実験	26
4.1.4	考察	30
4.2	SSF1.3	35
4.2.1	探索枝刈りによる高速化	35
4.2.2	計算機実験	39
4.2.3	考察	41
4.3	SSF1.4	46
4.3.1	探索本数削減による高速化	46
4.3.2	SSF1.4 の処理の概要	47
4.3.3	計算機実験	47
4.3.4	考察	50
4.4	まとめ	51
第 5 章	複素特異階段追跡法 (C-SSF)	56
5.1	C-SSF1.0	56
5.1.1	特異領域からの探索	57
5.1.2	C-SSF1.0 の処理の概要	57
5.1.3	計算機実験	58
5.1.4	考察	62
5.2	C-SSF1.1	68
5.2.1	探索枝刈りによる高速化	68
5.2.2	計算機実験	69
5.2.3	考察	71
5.3	C-SSF1.3	77

5.3.1	C-SSF1.3 の処理の概要	77
5.3.2	計算機実験	78
5.3.3	考察	80
5.4	まとめ	80
 第 6 章 SSF1.4 と C-SSF1.3 を用いた		
	カオス時系列の予測	86
6.1	ローレンツ方程式を用いた実験	87
6.2	2 重振り子を用いた実験	96
6.3	まとめ	109
 第 7 章 結論と展望		110
7.1	結論	110
7.2	展望	112
 謝 辞		113
 参考文献		114
 研究業績		117

目 次

2.1	多層パーセプトロン（バイアス有り）	5
2.2	条件数が大きい探索空間の概念図（黒の線：誤差の等高線，青のプラス：探索点，赤のアスタリスク：最適解）	10
2.3	局所最適解の概念図	11
2.4	特異領域の概念図	11
2.5	$\alpha\beta$ 写像を実 MLP($J = 2$) に適用した例	14
2.6	γ 写像を実 MLP($J = 2$) に適用した例	15
3.1	複素シグモイド関数	17
4.1	SSF1.2 の処理の概念図	23
4.2	特異領域上の固有ベクトルの概念図（固有ベクトルは実際には直交している。）	24
4.3	BP, BPQ, SSF1.2 の訓練誤差（人工データ）	28
4.4	BP, BPQ, SSF1.2 のテスト誤差（人工データ）	29
4.5	BP, BPQ, SSF1.2 の処理時間（秒）（人工データ）	30
4.6	BP, BPQ, SSF1.2 の重み更新平均回数（人工データ）	31
4.7	SSF1.2 の探索数（人工データ）	31
4.8	BP, BPQ, SSF1.2 の訓練誤差（実データ）	32
4.9	BP, BPQ, SSF1.2 のテスト誤差（実データ）	32
4.10	BP, BPQ, SSF1.2 の処理時間（秒）（実データ）	33
4.11	BP, BPQ, SSF1.2 の重み更新平均回数（実データ）	33
4.12	SSF1.2 の探索数（実データ）	34
4.13	探索枝刈りの概念図	35
4.14	探索枝刈りの概念図（軸は全て重み）	37

4.15 シグモイド関数	38
4.16 SSF1.3 の訓練誤差 (人工データ)	39
4.17 SSF1.3 のテスト誤差 (人工データ)	40
4.18 SSF1.3 の処理時間 (秒) (人工データ)	41
4.19 SSF1.3 の重み更新平均回数 (人工データ)	42
4.20 SSF1.3 の探索開始数 (実線) と最後まで探索枝刈りされなかった探索数 (点 線) (人工データ)	42
4.21 SSF1.3 の訓練誤差 (実データ)	43
4.22 SSF1.3 のテスト誤差 (実データ)	43
4.23 SSF1.3 の処理時間 (秒) (実データ)	44
4.24 SSF1.3 の重み更新平均回数 (実データ)	44
4.25 SSF1.3 の探索数 (実データ)	45
4.26 複数の負の固有値を持つ特異領域の概念図 (固有ベクトルは実際には直交 している.)	46
4.27 SSF1.4 の訓練誤差 (人工データ)	48
4.28 SSF1.4 のテスト誤差 (人工データ)	49
4.29 SSF1.4 の処理時間 (秒) (人工データ)	50
4.30 SSF1.4 の重み更新平均回数 (人工データ)	51
4.31 SSF1.4 の探索開始数 (実線) と最後まで探索枝刈りされなかった探索数 (点 線) (人工データ)	52
4.32 SSF1.4 の訓練誤差 (実データ)	53
4.33 SSF1.4 のテスト誤差 (実データ)	53
4.34 SSF1.4 の処理時間 (秒) (実データ)	54
4.35 SSF1.4 の重み更新平均回数 (実データ)	54
4.36 SSF1.4 の探索開始数 (実線) と最後まで探索枝刈りされなかった探索数 (点 線) (実データ)	55
5.1 C-BP, C-BFGS, C-SSF1.0 の訓練誤差 (人工データ 1)	59
5.2 C-BP, C-BFGS, C-SSF1.0 のテスト誤差 (人工データ 1)	60
5.3 C-BP, C-BFGS, C-SSF1.0 の処理時間 (秒) (人工データ 1)	61
5.4 C-BP, C-BFGS, C-SSF1.0 の重み更新平均回数 (人工データ 1)	62
5.5 C-SSF1.0 の探索回数 (人工データ 1)	63

5.6	C-BP, C-BFGS, C-SSF1.0 の訓練誤差 (人工データ 2)	63
5.7	C-BP, C-BFGS, C-SSF1.0 のテスト誤差 (人工データ 2)	64
5.8	C-BP, C-BFGS, C-SSF1.0 の処理時間 (秒) (人工データ 2)	64
5.9	C-BP, C-BFGS, C-SSF1.0 の重み更新平均回数 (人工データ 2)	65
5.10	C-BP, C-BFGS, C-SSF1.0 により得られた複素 MLP の出力 (人工データ 1)	66
5.11	C-SSF1.0 の探索回数 (人工データ 2)	67
5.12	C-SSF1.1 の訓練誤差 (人工データ 1)	70
5.13	C-SSF1.1 のテスト誤差 (人工データ 1)	71
5.14	C-SSF1.1 の処理時間 (秒) (人工データ 1)	72
5.15	C-SSF1.1 の重み更新平均回数 (人工データ 1)	72
5.16	C-SSF1.1 の探索開始数 (実線) と探索枝刈りされなかった探索回数 (点線) (人工データ 1)	73
5.17	C-SSF1.1 の訓練誤差 (人工データ 2)	73
5.18	C-SSF1.1 のテスト誤差 (人工データ 2)	74
5.19	C-SSF1.1 の処理時間 (秒) (人工データ 2)	74
5.20	C-SSF1.1 の重み更新平均回数 (人工データ 2)	75
5.21	C-SSF1.1 により得られた複素 MLP ($J = 16$) の出力 (人工データ 2)	75
5.22	C-SSF1.1 の探索開始数 (実線) と探索枝刈りされなかった探索回数 (点線) (人工データ 2)	76
5.23	C-SSF1.3 の訓練誤差 (人工データ 1)	78
5.24	C-SSF1.3 のテスト誤差 (人工データ 1)	79
5.25	C-SSF1.3 の処理時間 (秒) (人工データ 1)	80
5.26	C-SSF1.3 の重み更新平均回数 (人工データ 1)	81
5.27	C-SSF1.3 の探索開始数 (実線) と探索枝刈りされなかった探索回数 (点線) (人工データ 1)	82
5.28	C-SSF1.3 の訓練誤差 (人工データ 2)	83
5.29	C-SSF1.3 のテスト誤差 (人工データ 2)	83
5.30	C-SSF1.3 の処理時間 (秒) (人工データ 2)	84
5.31	C-SSF1.3 の重み更新平均回数 (人工データ 2)	84
5.32	C-SSF1.3 により得られた複素 MLP ($J = 12$) の出力 (人工データ 2)	85

5.33 C-SSF1.3の探索開始数（実線）と探索枝刈りされなかった探索回数（点線） （人工データ 2）	85
6.1 ローレンス方程式から生成された x, y, z の軌道（訓練データ生成領域） . .	88
6.2 訓練誤差（ローレンツ方程式）	90
6.3 テスト誤差（ローレンツ方程式）	91
6.4 BPQ のフリーラン予測（ローレンツ方程式）	92
6.5 SSF1.4 のフリーラン予測（ローレンツ方程式）	92
6.6 C-BFGS のフリーラン予測（ローレンツ方程式）	93
6.7 C-SSF1.3 のフリーラン予測（ローレンツ方程式）	93
6.8 処理時間（秒）（ローレンツ方程式）	94
6.9 重み更新平均回数（ローレンツ方程式）	95
6.10 2重振り子の初期位置	96
6.11 訓練誤差（2重振り子）	99
6.12 テスト誤差（2重振り子）	100
6.13 BPQ の2重振り子の θ のテストデータ予測結果	101
6.14 SSF1.4 の2重振り子の θ のテストデータ予測結果	101
6.15 C-BFGS の2重振り子の θ のテストデータ予測結果	102
6.16 C-SSF1.3 の2重振り子の θ のテストデータ予測結果	102
6.17 BPQ の2重振り子の θ のテストデータ予測結果	103
6.18 SSF1.4 の2重振り子の θ のテストデータ予測結果	103
6.19 C-BFGS の2重振り子の θ のテストデータ予測結果	104
6.20 C-SSF1.3 の2重振り子の θ のテストデータ予測結果	104
6.21 処理時間（秒）（2重振り子）	105
6.22 重み更新平均回数（2重振り子）	106
6.23 BPQ のフリーラン予測（2重振り子）	107
6.24 SSF1.4 のフリーラン予測（2重振り子）	107
6.25 C-BFGS のフリーラン予測（2重振り子）	108
6.26 C-SSF1.3 のフリーラン予測（2重振り子）	108

表 目 次

4.1	BP, BPQ, SSF1.2 の合計処理時間（人工データ）	28
4.2	BP, BPQ, SSF1.2 の合計処理時間（実データ）	30
4.3	SSF1.3 の合計処理時間（人工データ）	39
4.4	SSF1.3 の合計処理時間（実データ）	41
4.5	SSF1.4 の合計処理時間（人工データ）	48
4.6	SSF1.4 の合計処理時間（実データ）	50
5.1	C-SSF1.0 の合計処理時間（人工データ 1）	60
5.2	C-SSF1.0 の合計処理時間（人工データ 2）	63
5.3	C-SSF1.1 の合計処理時間（人工データ 1）	70
5.4	C-SSF1.1 の合計処理時間（人工データ 2）	71
5.5	C-SSF1.3 の合計処理時間（人工データ 1）	78
5.6	C-SSF1.3 の合計処理時間（人工データ 2）	80

第1章 序論

本章では，本論文の研究背景及び研究目的，本論文の構成について述べる．

1.1 研究背景

重みと入出力が全て実数の多層パーセプトロン (MLP: Multi-Layer Perceptron) は，理論上，隠れユニットが有限個あれば任意の非線形の問題を解ける [6] ため，関数近似やデータマイニング，画像認識など，幅広く応用されている．本論文では，重みと入出力が全て実数の MLP と，重みと入出力が全て複素数の複素 MLP を用いる．ここでは複素 MLP と区別するため，重みと入出力が全て実数の MLP を実 MLP と呼ぶ．

実 MLP の探索法としては勾配を用いて探索する BP(back propagation)[21] や勾配と Hesse 行列の逆行列の近似を用いて探索する準 Newton 法の一種の BPQ(Back Propagation based on Quasi-Newton) [22] などがある．本論文の計算機実験でも示すように，BPQ は BP よりも効率的に探索空間を降下できる．

しかし，実 MLP の重みが形成する探索空間には局所最適解が多数存在する [4] 問題がある．これは，隠れユニット群の並びや，活性化関数として用いられるシグモイド関数などの性質が一因である [27] が，それらを排除したとしても，複数の局所最適解が存在するとされる．そのため，BPQ を用いたとしても常に良質の解が得られるとは限らず，初期値を変えながら探索を複数回繰り返し，その中で最小の解を選択する方策が採られることが多い．しかし，初期値をどのように設定し，何回繰り返しせば良いか現在でも定説はない．

また，実 MLP の探索空間には勾配がゼロの連続領域である特異領域が存在する [5]．特

異領域は勾配がゼロであるため、探索が停滞してしまうことが知られている．探索空間にはこのような特異領域や Hesse 行列の条件数が大きい領域に満ちていることが指摘され [7]，実験的にも BPQ の探索では探索空間の条件数が 10^{15} を超えるような空間を通ることが報告されている [23]．そのため，特異領域を回避する方法が提案されてきた [1, 2] が，もし回避できたとしても，その先良質の解を得られる保証はない．そのため，常に安定して良質の解が得られる探索法が求められている．

複素 MLP は複素数の情報を自然に扱える上，実 MLP にはない特長を有するため，様々な応用が期待されている [8]．複素 MLP の活性化関数は，複素数をそのまま入力する fully complex 型と，複素数を実部と虚部に分けてからシグモイド関数などに入力する split complex 型があるが，本稿では複素数をそのままシグモイド関数に入力する fully complex 型の活性化関数 [10, 11] を用いる．この活性化関数は振幅が可変な周期関数など，複雑な関数をたった一つの隠れユニットで表現できるため，うまく学習すれば強力なモデルとなりえる．

複素 MLP を学習する方法としては，探索空間の勾配を用いる複素バックプロパゲーション法 (C-BP) [16, 17] や，勾配と Hesse 行列の逆行列の近似を用いる準 Newton 法の一種の複素 BFGS (C-BFGS) [28] などがある．C-BFGS は C-BP よりも効率良く探索空間を降下できるが，複素 MLP の探索空間には，実 MLP と同様，勾配がゼロの特異領域や局所最適解が多数存在する問題がある [18]．そのため，C-BFGS を用いたとしても常に良質の解が得られるとは限らず，実 MLP と同様，常に安定して良質の解が得られる探索法が求められている．

1.2 研究目的

本稿では，学習が停滞する特異領域を回避するのではなく，逆に利用する探索法である特異階段追跡法 (SSF: singularity stairs following) 1.2 を提案する．隠れユニットが J 個の実 MLP (実 MLP(J)) の特異領域は実 MLP($J - 1$) の最適解に可約性写像を適用すると形成される [5]．可約性写像により形成される特異領域上のほとんどの点は降下するルートが存在する鞍点である．また，このように形成された特異領域上の実 MLP(J) の入出力は実 MLP($J - 1$) の解と等価である．これらの性質を利用し，SSF1.2 では実 MLP($J - 1$) の最適解に可約性写像を適用して形成した実 MLP(J) の特異領域から探索空間を降下する．そのため，SSF1.2 は隠れユニットの増加に伴って訓練誤差が単調減少させることが

できる.

しかし, 探索空間の次元数は隠れユニットの増加に伴って増加し, その分特異領域も増えるため, 隠れユニットの増加に伴って処理時間が増大する問題がある. そのため, 現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を導入した SSF1.3 を提案する.

更に, 特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し, 探索数の上限を設定する手法により一層高速化した SSF1.4 を提案する. SSF1.4 では, 初期値をランダムに設定し複数回探索する従来の方法よりも高速に, かつ安定して良質の解が得られることが期待できる. 計算機実験では人工データ一種と実データ一種を用いて提案法を評価する.

複素 MLP の探索法としては, SSF1.2, SSF1.3, SSF1.4 をそれぞれ複素 MLP に拡張した C-SSF1.0, C-SSF1.1, C-SSF1.3 を提案する. C-SSF1.3 は SSF1.4 と同様, 従来法よりも高速に, かつ安定して良質の解が得られることが期待できる. 計算機実験では, 複素 MLP から生成した人工データと対数螺旋の線形和から生成した人工データを用いて提案法を評価する.

また, ある系の状態遷移規則が決定論的であるにも関わらず, 複雑な挙動を生み出す現象は決定論的カオスと呼ばれる [9]. 本論文では, 決定論的カオス挙動を示すローレンツ方程式 [13] と 2 重振り子 [26] から生成される時系列データの予測を, SSF1.4 と C-SSF1.3 により得られた解を用いて行い, SSF1.4 と C-SSF1.3 を評価する計算機実験も行う.

1.3 論文構成

本論文は以下のような構成である.

第2章では入出力と重みが全て実数の実多層パーセプトロンについて述べる. 特に第4章で提案する SSF において重要となる実 MLP の特異領域に重点を置く.

第3章では入出力と重みが全て複素数の複素多層パーセプトロンについて述べる. 複素 MLP の学習では勾配を用いるため, 非正則である目的関数を複素数の重みで微分する必要がある. 本論文では勾配を Wirtinger 微分を用いて勾配を求めるため, Wirtinger 微分についても述べる. また, 第5章で提案する C-SSF において特に重要となる複素 MLP の特異領域についても述べる.

第4章では, 学習が停滞する特異領域を逆に利用した SSF1.2 を提案する. その後, 探

索枝刈り手法を SSF1.2 に導入して高速化した SSF1.3 を提案する．更に，特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し，探索数の上限を設定する手法により一層高速化した SSF1.4 を提案する．評価実験で用いるデータとしては人工データ一種と実データ一種とする．

第 5 章では，実 MLP の探索法である SSF1.2, SSF1.3, SSF1.4 をそれぞれ複素 MLP の学習に拡張した C-SSF1.0, C-SSF1.1, C-SSF1.3 を提案する．評価実験で用いるデータとしては，複素 MLP から生成した人工データと対数螺旋の線形和から生成した人工データを用いる．

第 6 章では，第 4 章で提案した SSF1.4 と第 5 章で提案した CSSF1.3 を用い，決定論的カオス挙動を示すローレンツ方程式 [13] と 2 重振り子 [26] から生成される時系列データの時系列の予測を行う．

最後の第 7 章では，本論文のまとめと今後の展望について述べる．

第2章 実多層パーセプトロン(実MLP)

この章では重みと入出力が全て実数の MLP について説明する．実数の MLP は分類問題や回帰問題を解くことができ，理論上どのような関数も有限の隠れユニット数で分類，近似できる [6]．本論文では簡単のため出力ユニットは一つとする．また第3章で説明する複素数の MLP と区別するため，ここでは実数の MLP を実 MLP と呼ぶこととする．

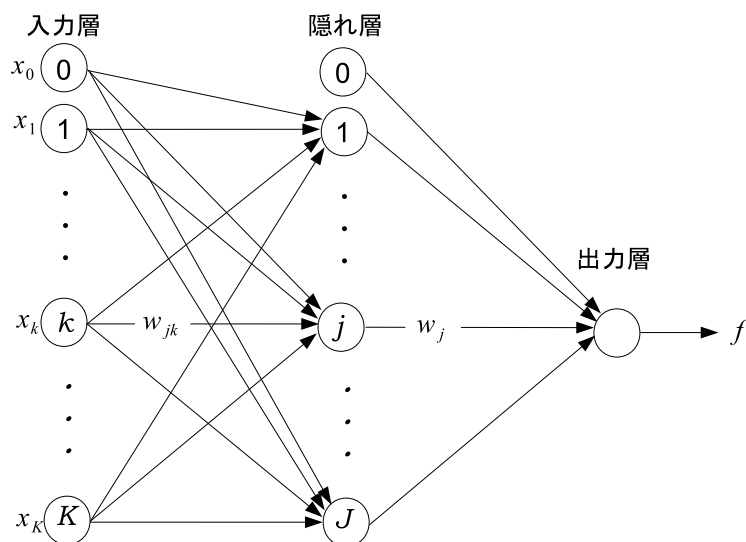


図 2.1: 多層パーセプトロン (バイアス有り)

2.1 実 MLP の出力

出力ユニットが一つの実 MLP の出力は以下である．ただし， $\mathbf{x} = (x_0, x_1, \dots, x_K)^T$ は入力ベクトル， $\boldsymbol{\theta}$ は重みの集合 $\{w_0, w_j, \mathbf{w}_j (= (w_{j0}, w_{j1}, \dots, w_{jK})^T), j = 1, \dots, J\}$ ， h は活性化関数，右肩の T は転置とする．

$$f(\mathbf{x}; \boldsymbol{\theta}) = w_0 + \sum_{j=1}^J w_j h(\mathbf{w}_j^T \mathbf{x}) \quad (2.1)$$

活性化関数としては以下のシグモイド関数がよく用いられる．

$$\sigma(a) = \frac{1}{1 + \exp(-a)} \quad (2.2)$$

2.2 実 MLP の学習法

学習法としては全てのサンプルを利用するバッチ型やサンプル毎に重みを更新するオンライン型などがあるが，本論文では全てバッチ型とする．

2.2.1 目的関数

本論文では 2 乗和誤差を目的関数として用いる．入力信号とその入力信号に対応する教師信号の対の集合である学習データ $\{(\mathbf{x}^\mu, y^\mu), \mu = 1, \dots, N\}$ が与えられたときの 2 乗和誤差は以下である．

$$E(\boldsymbol{\theta}) = \frac{1}{2} \sum_{\mu=1}^N (f(\mathbf{x}^\mu; \boldsymbol{\theta}) - y^\mu)^2 \quad (2.3)$$

2.2.2 BP

上記の目的関数を最急降下法により最小化する方法を BP(back propagation) と呼ぶ [21]．BP の重みの更新式を以下に示す．

$$\boldsymbol{\theta}^{(t+1)} \rightarrow \boldsymbol{\theta}^{(t)} - \eta \mathbf{g}^{(t)}, \quad \mathbf{g}^{(t)} \leq \frac{\partial E(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}^{(t)}} \quad (2.4)$$

学習率 η (探索幅) は定数とする方法や徐々に小さくする方法などがあるが，本稿では 2.3.1 で説明する直線探索の一種のバックトラッキング法を用いる．

2.2.3 Newton 法

BP は直線探索法を用いることで訓練誤差を単調減少させることができるが、2.3.1 で説明する条件数が大きいと学習が極端に遅くなってしまう。Newton 法は勾配のみでなく、目的関数の 2 階微分行列（Hesse 行列）も用いて探索を行う。

ここで、 $E(\boldsymbol{\theta})$ を探索点 $\boldsymbol{\theta}^{(t)}$ の周りで 2 次 Taylor 展開すると以下となる。

$$E(\boldsymbol{\theta}) \leftarrow E(\boldsymbol{\theta}^{(t)}) + \mathbf{g}^{(t)\top}(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)})^\top \mathbf{H}^{(t)}(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t)}), \quad \mathbf{H}^{(t)} \leq \frac{\partial^2 E(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}^{(t)} \partial \boldsymbol{\theta}^{(t)\top}} \quad (2.5)$$

上記の近似式を最小とする探索点 $\boldsymbol{\theta}$ は以下となる。

$$\boldsymbol{\theta} = \boldsymbol{\theta}^{(t)} - \mathbf{H}^{(t)-1} \mathbf{g}^{(t)} \quad (2.6)$$

そのため、Newton 法では以下のように重みを更新していく。

$$\boldsymbol{\theta}^{(t+1)} \rightarrow \boldsymbol{\theta}^{(t)} - \mathbf{H}^{(t)-1} \mathbf{g}^{(t)} \quad (2.7)$$

しかし、Newton 法では Hesse 行列を計算する必要があるため、それだけ手間がかかる。また、Hesse 行列の固有値に負のものが含まれることがあるため、その固有値に対応した固有ベクトル方向では逆に極大点に向かってしまう。

2.2.4 準 Newton 法

準 Newton 法は Hesse 行列の逆行列を近似しながら探索する方法である [25]。Hesse 行列の逆行列を近似する方法としては BFGS (Broyden-Fletcher-Goldfarb-Shanno method) がある。BFGS 法の更新式は以下である。ただし、 $\Delta \mathbf{g}^{(t)} = \mathbf{g}^{(t+1)} - \mathbf{g}^{(t)}$, $\Delta \boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}$ とする。

$$\mathbf{G}^{(t+1)} = \mathbf{G}_t + \left(1 + \frac{\Delta \mathbf{g}^{(t)\top} \mathbf{G}_t \Delta \mathbf{g}^{(t)}}{\Delta \mathbf{g}^{(t)\top} \Delta \boldsymbol{\theta}^{(t)}} \right) \frac{\Delta \boldsymbol{\theta}^{(t)} \Delta \boldsymbol{\theta}^{(t)\top}}{\Delta \mathbf{g}^{(t)\top} \Delta \boldsymbol{\theta}^{(t)}} + \frac{\Delta \boldsymbol{\theta}^{(t)} \Delta \mathbf{g}^{(t)\top} \mathbf{G}_t + \Delta \mathbf{G}_t \Delta \mathbf{g}^{(t)} \Delta \boldsymbol{\theta}^{(t)\top}}{\Delta \mathbf{g}^{(t)\top} \Delta \boldsymbol{\theta}^{(t)}} \quad (2.8)$$

以下の計算機実験では、 $\mathbf{G}^{(t)}$ の初期値 $\mathbf{G}^{(0)}$ は単位行列とした。

■ BPQ

Hesse 行列の逆行列を小メモリ BFGS で近似し、直線探索を以下の 2.2.5 で説明する 2 次の Taylor 展開を用いて求める方法を BPQ(Back Propagation based on Quasi-Newton) と呼ぶ [22]. 第 4, 6 章の計算機実験では小メモリ BFGS ではなく、式 (2.8) を用いるが、単に BPQ と呼ぶこととする.

2.2.5 直線探索

BP などの探索法では探索幅 η の値を固定する方法では誤差が増加する場合がある. そのため、以下では探索方向に直線探索を行って適切な η を求めることを考える. ここで、BP, 準 Newton 法の探索方向 $\mathbf{d}^{(t)}$ はそれぞれ $\mathbf{g}^{(t)}$, $\mathbf{G}^{(t)}\mathbf{g}^{(t)}$ である. 以下ではバクトラッキング法 [19] と 2 次の Taylor 展開により近似して最適探索幅を求める方法 [22] を説明する.

■ バクトラッキング法

バクトラッキング法の処理手順を以下に示す.

バクトラッキング法の処理の流れ

- 1: $\eta^{(0)}$ を 0 以上の値に設定し, $t_\eta = 0$ とする.
- 2: 終了条件を満たしたら処理を終了する.
- 3: $\eta^{(t_\eta+1)} \rightarrow \alpha\eta^{(t_\eta)}, t_\eta \rightarrow t_\eta + 1$ とし, 2 へ移動する.

本論文では一貫して $\eta^{(0)} = 1, \alpha = 0.5$ とし、終了条件は $E(\boldsymbol{\theta}^{(t)} + \eta^{(t_\eta)}\mathbf{d}^{(t)}) < E(\boldsymbol{\theta}^{(t)})$ となるか、 $\eta^{(t_\eta)}$ が 10^{-16} 以下となったときとした.

■ 2 次の Taylor 展開

ここでは、2 次の Taylor 展開により近似して最適探索幅を求める方法 [22] を考える. 現在点 $\mathbf{w}^{(t)}$ の $\mathbf{d}^{(t)}$ 方向の探索空間を $\psi(\eta) \leq E(\boldsymbol{\theta}^{(t)} + \eta\mathbf{d}^{(t)})$ とし、 $\psi(\eta)$ を $\eta = 0$ 周りで 2 次

Taylor 展開すると以下となる.

$$\psi(\eta) \leftarrow \psi(0) + \psi'(0)\eta + \frac{1}{2}\psi''(0)\eta^2, \quad \psi'(\eta) \leq \frac{d\psi(\eta)}{d\eta}, \quad \psi''(\eta) \leq \frac{d^2\psi(\eta)}{d\eta^2} \quad (2.9)$$

上記式を最小化する η は以下である.

$$\eta = \frac{\psi'(0)}{\psi''(0)} \quad (2.10)$$

上記式により η を求めることができるが, $\psi''(0)$ が正となるとは限らないため例外処理が必要となる. また, 2 次近似で実 MLP の探索空間を完全に近似することはできないため, 上記式により求めた η で誤差が減少しなければバクトラッキング法を適用するなどの処理が必要である.

2.3 実 MLP の問題点

実 MLP の問題として, 以下で説明するように, 実 MLP の探索空間には条件数が大きい領域や, 局所最適解や特異領域が存在することがある.

2.3.1 条件数

実 MLP の探索空間には, 2.3.3 で説明する特異領域のように Hesse 行列の固有値に 0 を含む領域や, 固有値が非常に小さい領域が存在する. そのため, そのような領域では以下に示す条件数 c は巨大となる. ただし, $\lambda_1, \lambda_2, \dots, \lambda_M$ は Hesse 行列の固有値, $|\lambda_1| \approx |\lambda_2| \approx \dots \approx |\lambda_M|$, M はパラメータ数とする.

$$c = \frac{|\lambda_1|}{|\lambda_M|} \quad (2.11)$$

そのため, 図 2.2 に示すような探索点に位置している場合, 勾配のみを利用する BP では最適解の方向を向くことができないため, 学習が困難となる.

2.3.2 局所最適解

実 MLP の探索空間には図 2.3 に示すような局所最適解が複数存在するという問題がある. そのため, 実 MLP の学習では重みの初期値によって異なる解に到達することがある.

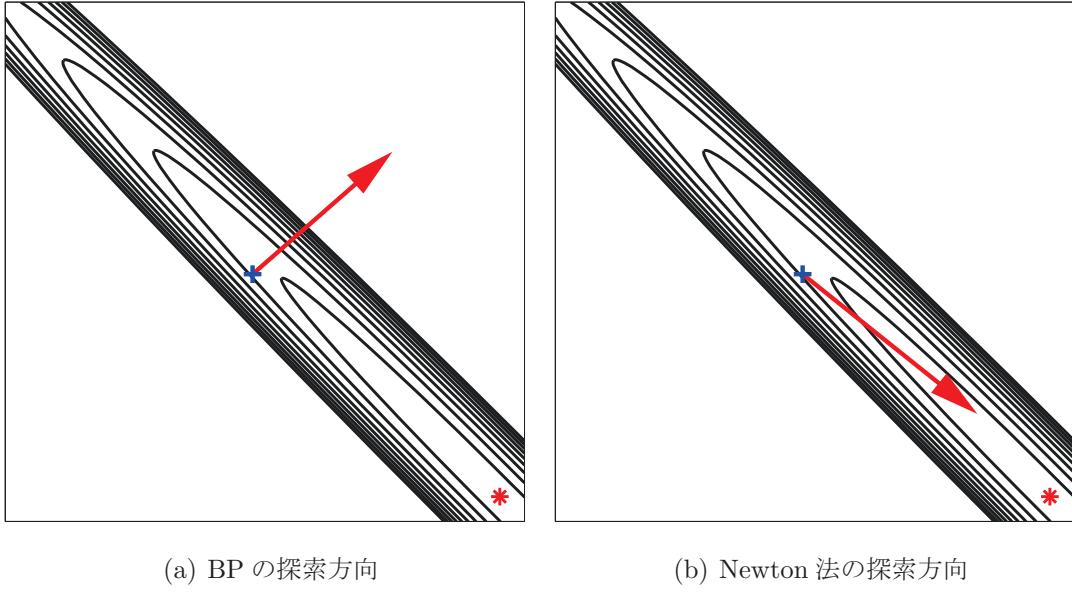


図 2.2: 条件数が大きい探索空間の概念図（黒の線：誤差の等高線，青のプラス：探索点，赤のアスタリスク：最適解）

ため，常に良質の解が得られる訳ではない．そのため初期値を変えて複数回繰り返し，得られた解から最良のものを選択する方法が採られている．

2.3.3 特異領域

実 MLP の探索空間には勾配がゼロの連続領域である特異領域が存在する．特異領域は勾配がゼロであるため，探索法は停滞してしまう．隠れユニットが J 個の実 MLP の特異領域は隠れユニットが $J - 1$ 個の実 MLP の最適解に以下で説明する可約性写像を適用すると形成される [5]．

■ 可約性写像

ここでは，隠れユニットが J 個の実 MLP を実 MLP(J) と表記する．また， $\boldsymbol{\theta}^{(J-1)} = \{u_0, u_j, \mathbf{u}_j, j = 2, \dots, J\}$ は実 MLP($J - 1$) の最適解とする． $\boldsymbol{\theta}^{(J-1)}$ は以下を満たす．

$$\frac{\partial E(\boldsymbol{\theta}^{(J-1)})}{\partial \boldsymbol{\theta}^{(J-1)}} = \mathbf{0} \quad (2.12)$$

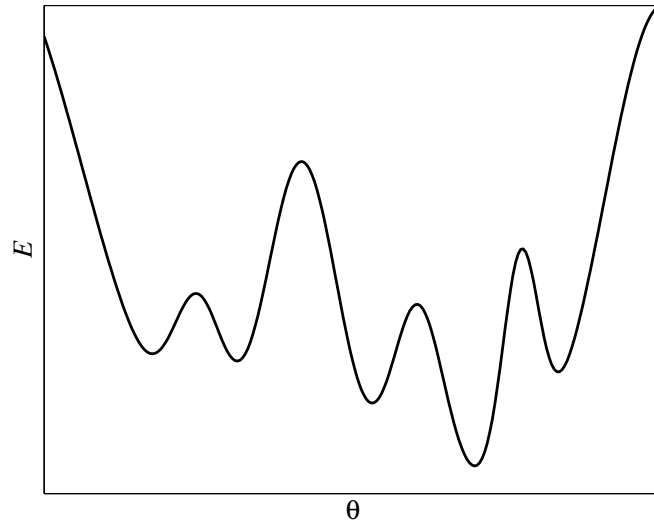


図 2.3: 局所最適解の概念図

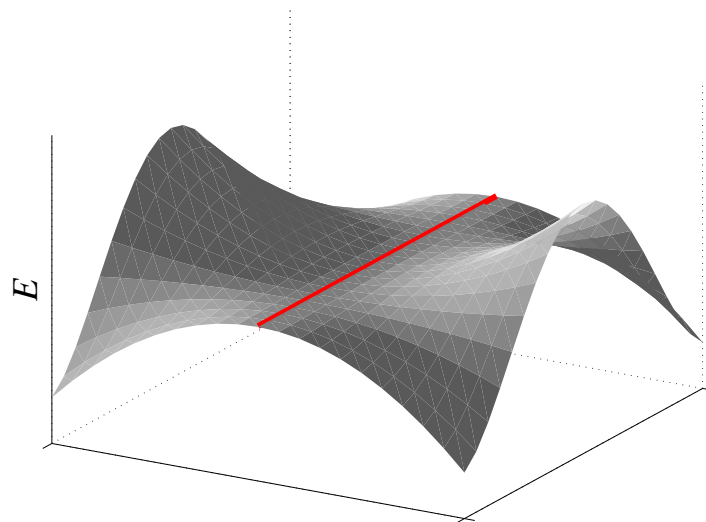


図 2.4: 特異領域の概念図

上記式をパラメータごとに示すと以下となる．ただし， $j = 2, \infty, J$ とする．

$$\begin{aligned} \frac{\partial E(\boldsymbol{\theta}^{(J-1)})}{\partial u_0} &= \bigwedge_{\mu=1}^N (f(\mathbf{x}^\mu; \boldsymbol{\theta}^{(J-1)}) - y^\mu) = 0 \\ \frac{\partial E(\boldsymbol{\theta}^{(J-1)})}{\partial u_j} &= \bigwedge_{\mu=1}^N (f(\mathbf{x}^\mu; \boldsymbol{\theta}^{(J-1)}) - y^\mu) h(\mathbf{u}_j^T \mathbf{x}^\mu) = 0 \\ \frac{\partial E(\boldsymbol{\theta}^{(J-1)})}{\partial \mathbf{u}_j} &= \bigwedge_{\mu=1}^N u_j (f(\mathbf{x}^\mu; \boldsymbol{\theta}^{(J-1)}) - y^\mu) h'(\mathbf{u}_j^T \mathbf{x}^\mu) \mathbf{x}^\mu = \mathbf{0} \end{aligned}$$

可約性写像は α, β, γ の 3 種類がある． $\boldsymbol{\theta}^{(J-1)}$ に 3 種の可約性写像を適用して得られる領域を以下に示す．ただし，それぞれ $\boldsymbol{\Theta}_\alpha^{(J)}, \boldsymbol{\Theta}_\beta^{(J)}, \boldsymbol{\Theta}_\gamma^{(J)}$ とする [15]．

$$\boldsymbol{\theta}^{(J-1)} \xrightarrow{\alpha} \boldsymbol{\Theta}_\alpha^{(J)}, \quad \boldsymbol{\theta}^{(J-1)} \xrightarrow{\beta} \boldsymbol{\Theta}_\beta^{(J)}, \quad \boldsymbol{\theta}^{(J-1)} \xrightarrow{\gamma} \boldsymbol{\Theta}_\gamma^{(J)}$$

$$\boldsymbol{\Theta}_\alpha^{(J)} \leq \{ \boldsymbol{\theta}^{(J)} \setminus w_0 = u_0, w_1 = 0, w_j = u_j, \mathbf{w}_j = \mathbf{u}_j, j = 2, \infty, J \} \quad (2.13)$$

$$\begin{aligned} \boldsymbol{\Theta}_\beta^{(J)} \leq \{ \boldsymbol{\theta}^{(J)} \setminus w_0 + w_1 g(w_{10}) = u_0, \mathbf{w}_1 = (w_{10}, 0, \infty, 0)^T, \\ w_j = u_j, \mathbf{w}_j = \mathbf{u}_j, j = 2, \infty, J \} \end{aligned} \quad (2.14)$$

$$\begin{aligned} \boldsymbol{\Theta}_\gamma^{(J)} \leq \{ \boldsymbol{\theta}^{(J)} \setminus w_0 = u_0, w_1 + w_m = u_m, \mathbf{w}_1 = \mathbf{w}_m = \mathbf{u}_m, \\ w_j = u_j, \mathbf{w}_j = \mathbf{u}_j, j \in \{ 2, \infty, J \} \} \quad (2.15) \end{aligned}$$

ただし， $m = 2, \infty, J$ とする．

領域 $\boldsymbol{\Theta}_\alpha^{(J)}, \boldsymbol{\Theta}_\beta^{(J)}, \boldsymbol{\Theta}_\gamma^{(J)}$ 上の $\boldsymbol{\theta}^{(J)}$ は $\boldsymbol{\theta}^{(J-1)}$ と出力が等価であるため， $\boldsymbol{\theta}^{(J)}$ は $\boldsymbol{\theta}^{(J-1)}$ に可約である．

■ 可約性写像により形成される特異領域

可約性写像を適用して形成された領域の中には、以下を満たす、勾配がゼロとなる連続領域が存在する。

$$\frac{\partial E(\boldsymbol{\theta}^{(J)})}{\partial w_0} = \bigwedge_{\mu} (f(\mathbf{x}^{\mu}; \boldsymbol{\theta}^{(J)}) - y^{\mu}) = 0 \quad (2.16)$$

$$\frac{\partial E(\boldsymbol{\theta}^{(J)})}{\partial w_1} = \bigwedge_{\mu} (f(\mathbf{x}^{\mu}; \boldsymbol{\theta}^{(J)}) - y^{\mu}) h(\mathbf{w}_1^T \mathbf{x}^{\mu}) = 0 \quad (2.17)$$

$$\frac{\partial E(\boldsymbol{\theta}^{(J)})}{\partial w_j} = \bigwedge_{\mu} (f(\mathbf{x}^{\mu}; \boldsymbol{\theta}^{(J)}) - y^{\mu}) h(\mathbf{w}_j^T \mathbf{x}^{\mu}) = 0, \quad (2.18)$$

$$\frac{\partial E(\boldsymbol{\theta}^{(J)})}{\partial w_1} = \bigwedge_{\mu} (f(\mathbf{x}^{\mu}; \boldsymbol{\theta}^{(J)}) - y^{\mu}) h'(\mathbf{w}_1^T \mathbf{x}^{\mu}) \mathbf{x}^{\mu} = \mathbf{0} \quad (2.19)$$

$$\frac{\partial E(\boldsymbol{\theta}^{(J)})}{\partial w_j} = \bigwedge_{\mu} (f(\mathbf{x}^{\mu}; \boldsymbol{\theta}^{(J)}) - y^{\mu}) h'(\mathbf{w}_j^T \mathbf{x}^{\mu}) \mathbf{x}^{\mu} = \mathbf{0} \quad (2.20)$$

領域 $\boldsymbol{\Theta}_J^{\alpha}, \boldsymbol{\Theta}_J^{\beta}, \boldsymbol{\Theta}_J^{\gamma}$ では、 $f(\mathbf{x}^{\mu}; \boldsymbol{\theta}^{(J)}) = f(\mathbf{x}^{\mu}; \boldsymbol{\theta}^{(J-1)})$ が成り立ち、かつ $h(\mathbf{w}_j^T \mathbf{x}^{\mu}) = h(\mathbf{u}_j^T \mathbf{x}^{\mu})$, $j = 2, \dots, J$ である。そのため、式(2.16), (2.18), (2.20)は成立しているが、式(2.17), (2.19)が成立しているかはそれぞれの領域により異なる。

- (1) 領域 $\boldsymbol{\Theta}_J^{\alpha}$ では、重み $\mathbf{w}_1$ が全て自由であるため、式(2.17)が成り立つ保証はない。
- (2) 領域 $\boldsymbol{\Theta}_J^{\beta}$ では、 $h(\mathbf{w}_1^T \mathbf{x}^{\mu}) = h(w_{10})$ となり、サンプル μ に依存しないため、式(2.17)は成立する。しかし、一般に、式(2.19)が成り立つ保証はなく、成立させるには、 $w_1 = 0$ である必要がある。そのため、領域 $\boldsymbol{\Theta}_{\alpha}^{(J)}$ と $\boldsymbol{\Theta}_{\beta}^{(J)}$ が重なる領域は w_{10} が自由である1次元の特異領域である。以下が成立するこの領域を $\boldsymbol{\Theta}_{\alpha\beta}^{(J)}$ とする。

$$\begin{aligned} w_0 &= u_0, \quad w_1 = 0, \quad \mathbf{w}_1 = (w_{10}, 0, \dots, 0)^T \\ w_j &= u_j, \quad \mathbf{w}_j = \mathbf{u}_j, \quad j = 2, \dots, J \end{aligned} \quad (2.21)$$

- (2) 領域 $\boldsymbol{\Theta}_{\gamma}^{(J)}$ は、 $h(\mathbf{w}_1^T \mathbf{x}^{\mu}) = h(\mathbf{u}_m^T \mathbf{x}^{\mu})$ であるため、式(2.17), (2.19)はどちらも成立する。そのため、領域 $\boldsymbol{\Theta}_{\gamma}^{(J)}$ は以下の式を満たす1次元の特異領域である。

$$w_1 + w_m = u_m \quad (2.22)$$

以上説明したように、領域 $\boldsymbol{\Theta}_{\alpha\beta}^{(J)}, \boldsymbol{\Theta}_{\gamma}^{(J)}$ は特異領域であり、これらの特異領域上のほと

んどの点は Hesse 行列の固有値に負を含むため鞍点である [5]. 図 2.5, 2.6 に入力数が 2, $J = 2$ の実 MLP の解に $\alpha\beta$ 写像を適用したときと, γ 写像を適用したときの様子を示す.

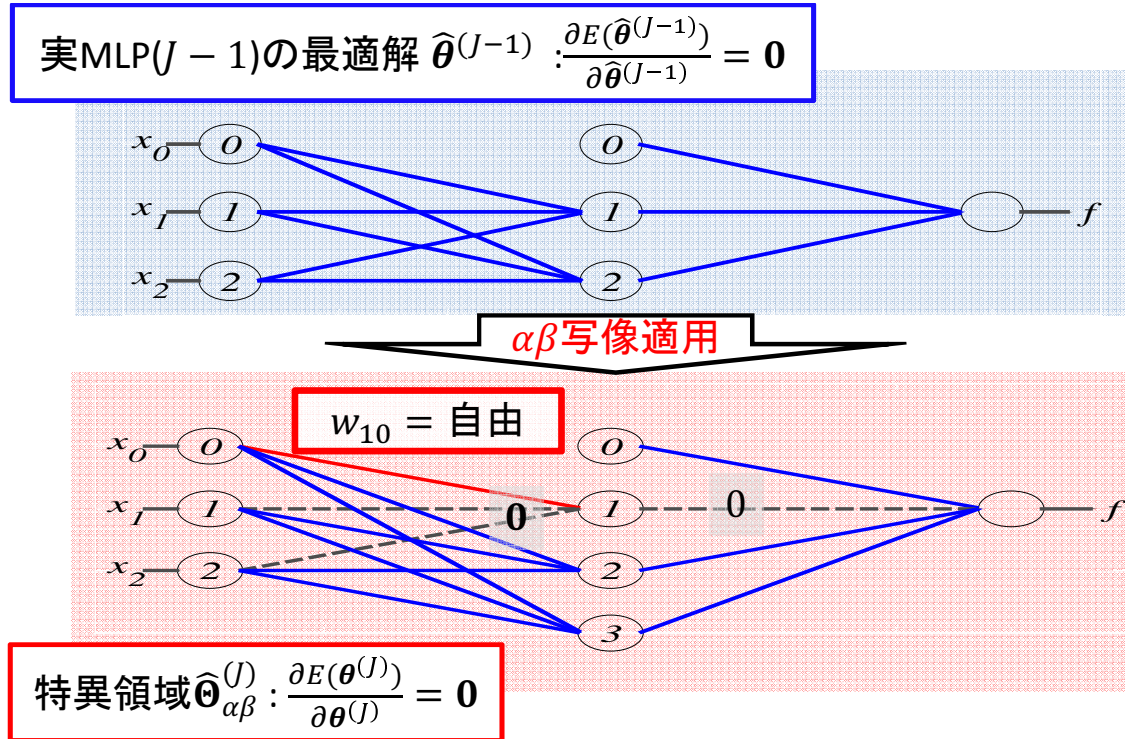


図 2.5: $\alpha\beta$ 写像を実 MLP($J = 2$) に適用した例

第3章 複素多層パーセプトロン (複素MLP)

本章では入出力と重みが複素数の複素MLPについて説明する．複素MLPは複素数の情報を自然に扱える上，実MLPにない特長を有するため，様々な応用が期待されている[8]．また，第2章と同様，出力ユニットは1つとする．

3.1 複素MLPの出力

出力ユニットが1つの複素MLPの出力は以下である．ただし， $\mathbf{x} = (x_0, x_1, \dots, x_K)^T$ は入力ベクトル， $\boldsymbol{\theta}_z$ は重みの集合 $\{w_0, w_j, \mathbf{w}_j = (w_{j0}, w_{j1}, \dots, w_{jK})^T, j = 1, \dots, J\}$ ， h は活性化関数，右肩の T は転置とする．

$$f(\mathbf{x}; \boldsymbol{\theta}_z) = w_0 + \sum_{j=1}^J w_j h(\mathbf{w}_j^T \mathbf{x}) \quad (3.1)$$

上記式は実MLPの出力の式とほぼ同じであるが，入出力と重みが全て複素数である点異なる．

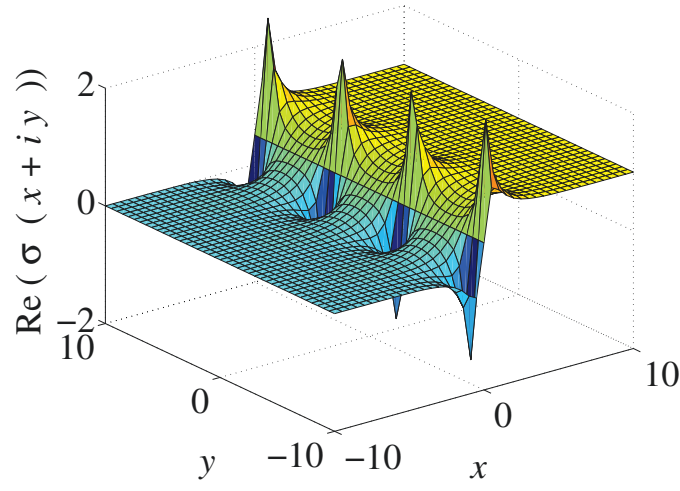
3.1.1 複素MLPの活性化関数

複素MLPの活性化関数には，複素数をそのままシグモイド関数などに入力する fully complex 型と，複素数を実部と虚部に分けてからシグモイド関数などに入力する split com-

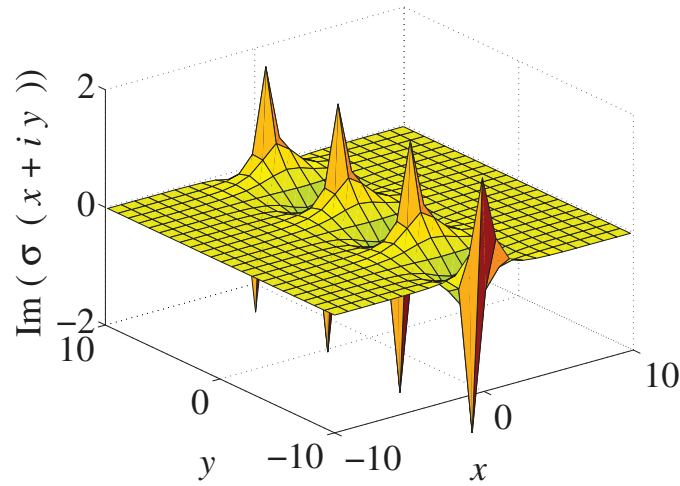
plex 型がある．本論文では一貫して以下のシグモイド関数に複素数をそのまま入力する fully complex 型の活性化関数を用いる [10, 11]．ただし， $c = a + ib$ ， $i = \sqrt{-1}$ とする．

$$\sigma(c) = \frac{1}{1 + e^{-c}} = \frac{1 + e^{-a} \cos b + ie^{-a} \sin b}{1 + 2e^{-a} \cos b + e^{-2a}} \quad (3.2)$$

ここではこの活性化関数を複素シグモイド関数と呼ぶ．図 3.1 に複素シグモイド関数を示す．複素シグモイド関数は特異点を含む正則関数であるが，振幅が可変な周期関数など複



(a) 実部



(b) 虚部

図 3.1: 複素シグモイド関数

雑な関数を表現できるため，うまく学習すれば強力なモデルとなりえる．

3.2 複素 MLP の目的関数

複素 MLP の学習では、通常、以下のような目的関数を考える。ただし、入出力の対の集合 $\{(\mathbf{x}^\mu, y^\mu), \mu = 1, \dots, N\}$ は複素数の学習データとする。

$$E_c = \sum_{\mu=1}^N \delta^\mu \overline{\delta^\mu}, \quad \delta^\mu = f(\mathbf{x}^\mu; \boldsymbol{\theta}_z) - y^\mu \quad (3.3)$$

式 (3.3) の目的関数は非正則であるが、Wirtinger 微分 [3] を用いれば勾配や Hesse 行列が容易に計算できるため、本論文では、3.3 で説明する Wirtinger 微分を利用する。

複素 MLP に教師信号が実数の問題を適用する場合は、式 (3.3) の目的関数を用いると出力の虚部が 0 となるように重みが学習される。そこで、出力の実部のみを学習する以下の目的関数 [24] を考える。

$$E_r = \frac{1}{2} \sum_{\mu=1}^N |\delta^\mu + \overline{\delta^\mu}|^2 \quad (3.4)$$

目的関数をこのようにすれば、出力の虚部は任意とでき、実部をより強力に近似できるようになり、最適モデルの重みの総数を減少させられると考えられる。

3.3 Wirtinger 微分による勾配と Hesse 行列の計算

式 (3.3), (3.4) のような非正則である目的関数に対しては、Wirtinger 微分 [3] を用いることでその勾配や Hesse 行列が容易に計算できる。ここで、Wirtinger 微分における多変量の記法である $\mathbf{c}$ 表現, $\mathbf{z}$ 表現, $\mathbf{r}$ 表現で表した重みをそれぞれ $\boldsymbol{\theta}_c, \boldsymbol{\theta}_z, \boldsymbol{\theta}_r$ とする。 $\boldsymbol{\theta}_c, \boldsymbol{\theta}_z, \boldsymbol{\theta}_r$ の関係式を以下に示す。ただし, $\boldsymbol{\theta}_x = \text{Re}(\boldsymbol{\theta}_z), \boldsymbol{\theta}_y = \text{Im}(\boldsymbol{\theta}_z), \bar{\mathbf{z}}$ は $\mathbf{z}$ の複素共役, $\mathbf{I}$ は単位行列とする。

$$\boldsymbol{\theta}_c^T = \boldsymbol{\theta}_z^T, \overline{\boldsymbol{\theta}_z}^T \left(\begin{array}{l} \boldsymbol{\theta}_z^T = (\mathbf{I}, i\mathbf{I}) \boldsymbol{\theta}_r, \quad \boldsymbol{\theta}_r^T = \boldsymbol{\theta}_x^T, \boldsymbol{\theta}_y^T \end{array} \right) \quad (3.5)$$

Wirtinger 微分では $\mathbf{z}$ と $\bar{\mathbf{z}}$ が独立とみなせるため、以下の式が成り立つ。ただし, f は

実数のスカラーを出力する複素関数 $f = f(\mathbf{z}, \bar{\mathbf{z}})$, $\frac{\partial}{\partial \mathbf{a}} = (\frac{\partial}{\partial a_1}, \frac{\partial}{\partial a_2}, \dots)^T$ とする.

$$\mathbf{g}_z = \frac{\partial f}{\partial \boldsymbol{\theta}_z} = \frac{1}{2} \left(\frac{\partial f}{\partial \boldsymbol{\theta}_x} - i \frac{\partial f}{\partial \boldsymbol{\theta}_y} \right), \quad \bar{\mathbf{g}}_z = \frac{\partial f}{\partial \bar{\boldsymbol{\theta}}_z} = \frac{1}{2} \left(\frac{\partial f}{\partial \boldsymbol{\theta}_x} + i \frac{\partial f}{\partial \boldsymbol{\theta}_y} \right) \quad (3.6)$$

$$\mathbf{g}_r = \frac{\partial f}{\partial \boldsymbol{\theta}_r} = 2 \left(\frac{\partial f}{\partial \boldsymbol{\theta}_z} \right)^T \begin{pmatrix} \text{Re} \\ \text{Im} \end{pmatrix} \quad (3.7)$$

$$\mathbf{g}_c = \frac{\partial f}{\partial \boldsymbol{\theta}_c} = \left(\frac{\partial f}{\partial \boldsymbol{\theta}_z}^T, \frac{\partial f}{\partial \bar{\boldsymbol{\theta}}_z}^T \right)^T \quad (3.8)$$

$\mathbf{c}$ 表現の Hesse 行列 $\mathbf{H}_{\bar{\mathbf{c}}\mathbf{c}}$ と $\mathbf{r}$ 表現の Hesse 行列 $\mathbf{H}_{\mathbf{r}\mathbf{r}}$ は以下の関係にある. また, $\mathbf{a}^H = \bar{\mathbf{a}}^T$ とする.

$$\mathbf{H}_{\mathbf{r}\mathbf{r}} = \mathbf{J}^H \mathbf{H}_{\bar{\mathbf{c}}\mathbf{c}} \mathbf{J}, \quad \mathbf{J} = \begin{pmatrix} \mathbf{I}, & i\mathbf{I} \\ \mathbf{I}, & i\mathbf{I} \end{pmatrix} \quad (3.9)$$

ただし, 以下とする.

$$\mathbf{H}_{\mathbf{r}\mathbf{r}} = \frac{\partial^2 f}{\partial \boldsymbol{\theta}_r \partial \boldsymbol{\theta}_r^T}, \quad \mathbf{H}_{\bar{\mathbf{c}}\mathbf{c}} = \frac{\partial^2 f}{\partial \bar{\boldsymbol{\theta}}_c \partial \bar{\boldsymbol{\theta}}_c^H} = \begin{pmatrix} \mathbf{H}_{\bar{\mathbf{z}}\bar{\mathbf{z}}}, & \mathbf{H}_{\bar{\mathbf{z}}\mathbf{z}} \\ \mathbf{H}_{\mathbf{z}\bar{\mathbf{z}}}, & \mathbf{H}_{\mathbf{z}\mathbf{z}} \end{pmatrix} \quad (3.10)$$

$$\mathbf{H}_{\bar{\mathbf{z}}\bar{\mathbf{z}}} = \frac{\partial^2 f}{\partial \bar{\boldsymbol{\theta}}_z \partial \bar{\boldsymbol{\theta}}_z^H}, \quad \mathbf{H}_{\bar{\mathbf{z}}\mathbf{z}} = \frac{\partial^2 f}{\partial \bar{\boldsymbol{\theta}}_z \partial \boldsymbol{\theta}_z^H}, \quad (3.11)$$

$$\mathbf{H}_{\mathbf{z}\bar{\mathbf{z}}} = \frac{\partial^2 f}{\partial \boldsymbol{\theta}_z \partial \bar{\boldsymbol{\theta}}_z^H}, \quad \mathbf{H}_{\mathbf{z}\mathbf{z}} = \frac{\partial^2 f}{\partial \boldsymbol{\theta}_z \partial \boldsymbol{\theta}_z^H} \quad (3.12)$$

$\mathbf{r}, \mathbf{z}, \mathbf{c}$ 表現での $\boldsymbol{\theta}_z^{(t)}$ 周りの二次の Taylor 展開はそれぞれ以下となる.

$$f(\boldsymbol{\theta}_r) \leftarrow f(\boldsymbol{\theta}_r^{(t)}) + \mathbf{g}_r^{(t)T} (\boldsymbol{\theta}_r - \boldsymbol{\theta}_r^{(t)}) + \frac{1}{2} (\boldsymbol{\theta}_r - \boldsymbol{\theta}_r^{(t)})^T \mathbf{H}_{\mathbf{r}\mathbf{r}}^{(t)} (\boldsymbol{\theta}_r - \boldsymbol{\theta}_r^{(t)}) \quad (3.13)$$

$$f(\boldsymbol{\theta}_z) \leftarrow f(\boldsymbol{\theta}_z^{(t)}) + 2 \text{Re} \left(\mathbf{g}_z^{(t)T} (\boldsymbol{\theta}_z - \boldsymbol{\theta}_z^{(t)}) \right) + \text{Re} \left((\boldsymbol{\theta}_z - \boldsymbol{\theta}_z^{(t)})^H \mathbf{H}_{\bar{\mathbf{z}}\bar{\mathbf{z}}}^{(t)} (\boldsymbol{\theta}_z - \boldsymbol{\theta}_z^{(t)}) + (\boldsymbol{\theta}_z - \boldsymbol{\theta}_z^{(t)})^H \mathbf{H}_{\bar{\mathbf{z}}\mathbf{z}}^{(t)} (\boldsymbol{\theta}_z - \boldsymbol{\theta}_z^{(t)}) \right) \quad (3.14)$$

$$f(\boldsymbol{\theta}_c) \leftarrow f(\boldsymbol{\theta}_c^{(t)}) + \mathbf{g}_c^T (\boldsymbol{\theta}_c - \boldsymbol{\theta}_c^{(t)}) + \frac{1}{2} (\boldsymbol{\theta}_c - \boldsymbol{\theta}_c^{(t)})^H \mathbf{H}_{\bar{\mathbf{c}}\mathbf{c}}^{(t)} (\boldsymbol{\theta}_c - \boldsymbol{\theta}_c^{(t)}) \quad (3.15)$$

$\mathbf{H}_{\mathbf{r}\mathbf{r}}$ は実数対称行列, $\mathbf{H}_{\bar{\mathbf{c}}\mathbf{c}}$ は Hermite 行列であるため, 共に固有値は実数となる. また, $\mathbf{H}_{\mathbf{r}\mathbf{r}}$ の固有値は $\mathbf{H}_{\bar{\mathbf{c}}\mathbf{c}}$ の固有値の 2 倍の大きさとなる. そのため, $\mathbf{H}_{\mathbf{r}\mathbf{r}}$ と $\mathbf{H}_{\bar{\mathbf{c}}\mathbf{c}}$ の条件数は同じ値となる. 第 5 章で提案する探索法では $\mathbf{H}_{\mathbf{r}\mathbf{r}}$ の固有値と固有ベクトルを利用する.

3.4 複素 MLP の学習法

複素 MLP を学習する方法としては、探索空間の勾配を用いる複素バックプロパゲーション法 (C-BP) [16, 17] や、勾配と Hesse 行列の逆行列の近似を用いる準 Newton 法の一種の複素 BFGS (C-BFGS: Complex-valued Broyden-Fletcher-Goldfarb-Shanno) [28] などがある。複素 MLP の学習においても、実 MLP と同様、本論文では全てバッチ型とする。

3.4.1 C-BP

C-BP では、実 MLP の BP と同様、探索空間の勾配を用いる。C-BP の重みの更新式を以下に示す。

$$\boldsymbol{\theta}_z^{(t+1)} \rightarrow \boldsymbol{\theta}^{(t)} \quad \eta \mathbf{g}^{(t)}, \quad \mathbf{g}^{(t)} \leq \frac{\partial E(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}^{(t)}} \quad (3.16)$$

BP と同様、学習率 η は定数とする方法や徐々に小さくする方法などがあるが、本稿では 2.2.5 で説明したバックトラッキング法を用いて直線探索を行う。

3.4.2 C-BFGS

C-BFGS としては、 $\mathbf{r}$ 表現の複素 MLP の重みに BFGS を適用する方法 [28] と、 $\mathbf{c}$ 表現の重みに BFGS を適用する方法がある [20]。本稿では前者の方法を採用。そのため、式 (2.8) の BFGS 法の更新式をそのまま適用できる。また、直線探索としてはバックトラッキング法を用いる。

3.5 複素 MLP の特異領域

複素 MLP の探索空間上にも、実 MLP と同様、特異領域が存在する。複素 MLP の特異領域においても隠れユニット数が $J - 1$ 個の複素 MLP の最適解に可約性写像を適用して形成された特異領域のほとんどが鞍点であることが示されている [18]。

■ 可約性写像

ここでは隠れユニットが J 個の複素 MLP を複素 MLP(J) と表記する．複素 MLP の可約性写像も以下の 3 種類 α, β, γ が考えられる．複素 MLP($J-1$) の $\mathbf{z}$ 表現の最適解 $\boldsymbol{\theta}_{\mathbf{z}}^{(J-1)} = \{u_0, u_j, \mathbf{u}_j, j = 2, \dots, J\}$ に 3 種の可約性写像を適用して得られる領域をそれぞれ $\Theta_{\alpha}^{(J)}, \Theta_{\beta}^{(J)}, \Theta_{\gamma}^{(J)}$ とする [15]．ただし, $\Theta_{\alpha}^{(J)}, \Theta_{\beta}^{(J)}, \Theta_{\gamma}^{(J)}$ は $\mathbf{z}$ 表現とする．

$$\boldsymbol{\theta}_{\mathbf{z}}^{(J-1)} \xrightarrow{\alpha} \Theta_{\alpha}^{(J)}, \quad \boldsymbol{\theta}_{\mathbf{z}}^{(J-1)} \xrightarrow{\beta} \Theta_{\beta}^{(J)}, \quad \boldsymbol{\theta}_{\mathbf{z}}^{(J-1)} \xrightarrow{\gamma} \Theta_{\gamma}^{(J)}$$

$$\Theta_{\alpha}^{(J)} \leq \{\boldsymbol{\theta}_{\mathbf{z}}^{(J)} \setminus w_0 = u_0, w_1 = 0, w_j = u_j, \mathbf{w}_j = \mathbf{u}_j, j = 2, \dots, J\} \quad (3.17)$$

$$\Theta_{\beta}^{(J)} \leq \{\boldsymbol{\theta}_{\mathbf{z}}^{(J)} \setminus w_0 + w_1 g(w_{10}) = u_0, \mathbf{w}_1 = (w_{10}, 0, \dots, 0)^T, \\ w_j = u_j, \mathbf{w}_j = \mathbf{u}_j, j = 2, \dots, J\} \quad (3.18)$$

$$\Theta_{\gamma}^{(J)} \leq \{\boldsymbol{\theta}_{\mathbf{z}}^{(J)} \setminus w_0 = u_0, w_1 + w_m = u_m, \mathbf{w}_1 = \mathbf{w}_m = \mathbf{u}_m, \\ w_j = u_j, \mathbf{w}_j = \mathbf{u}_j, j \in \{2, \dots, J\} \setminus m\} \quad (3.19)$$

ただし, $m = 2, \dots, J$ とする．

可約性写像を適用し, 隠れユニットを一つ増やして形成した領域 $\Theta_{\alpha}^{(J)}, \Theta_{\beta}^{(J)}, \Theta_{\gamma}^{(J)}$ 上の点 $\boldsymbol{\theta}_{\mathbf{z}}^{(J)}$ は $\boldsymbol{\theta}_{\mathbf{z}}^{(J-1)}$ と出力が等価であるため, $\boldsymbol{\theta}_{\mathbf{z}}^{(J)}$ は $\boldsymbol{\theta}_{\mathbf{z}}^{(J-1)}$ に可約である．

■ 可約性写像により形成される特異領域

可約性写像を適用して形成された領域の中には勾配がゼロとなる領域が存在する．勾配がゼロの連続領域となる領域は以下である．

(1) 領域 $\Theta_{\alpha}^{(J)}$ と $\Theta_{\beta}^{(J)}$ が重なる領域は w_{10} が自由である． w_{10} は複素数であるため 2 次元の特異領域である．以下が成立するこの領域を $\Theta_{\alpha\beta}^{(J)}$ とする．

$$w_0 = u_0, \quad w_1 = 0, \quad \mathbf{w}_1 = (w_{10}, 0, \dots, 0)^T \\ w_j = u_j, \quad \mathbf{w}_j = \mathbf{u}_j, \quad j = 2, \dots, J \quad (3.20)$$

(2) 領域 $\Theta_{\gamma}^{(J)}$ は, 以下の式を満たす特異領域である．

$$w_1 + w_m = u_m \quad (3.21)$$

上記の重みは複素数であるため, この特異領域も 2 次元である．

第4章 特異階段追跡法 (SSF)

第2章で説明したように，可約性写像により形成される特異領域上のほとんどの点は降下するルートが存在する鞍点である．この性質を利用し，実 MLP の学習が停滞する特異領域を回避するのではなく，逆に利用する探索法である特異階段追跡法 (SSF: singularity stairs following) 1.2 を 4.1 で提案する．この方法は隠れユニットが $J - 1$ 個の実 MLP (実 $\text{MLP}(J - 1)$) の最適解に可約性写像を適用して形成した実 $\text{MLP}(J)$ の特異領域から探索空間を降下する．このように形成された特異領域上の実 $\text{MLP}(J)$ の入出力は実 $\text{MLP}(J - 1)$ の解と等価であるため，SSF1.2 は隠れユニットの増加に伴って訓練誤差が単調減少させることができる．

しかし，探索空間の次元数は隠れユニットの増加に伴って増加し，その分特異領域も増えるため，隠れユニットの増加に伴って処理時間が増大する問題がある．そのため 4.2 では，現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を導入した SSF1.3 を提案する．

更に 4.3 では，特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し，探索数の上限を設定する手法により一層高速化した SSF1.4 を提案する．

4.1 SSF1.2

SSF1.2 では，図 4.1 に示すように，実 $\text{MLP}(J - 1)$ の探索で得られた解に可約性写像を適用して形成した特異領域から実 $\text{MLP}(J)$ の探索を行う．

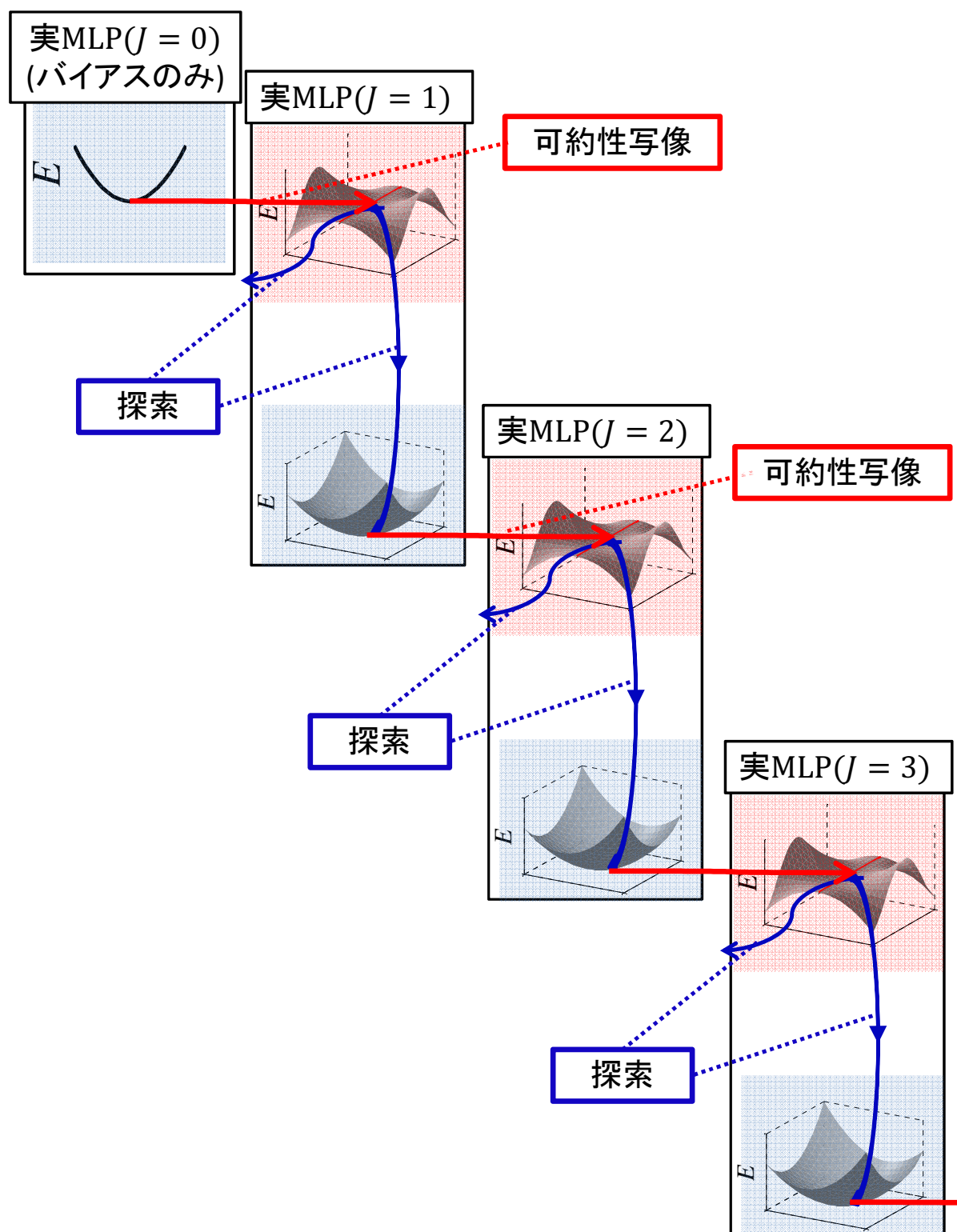


図 4.1: SSF1.2 の処理の概念図

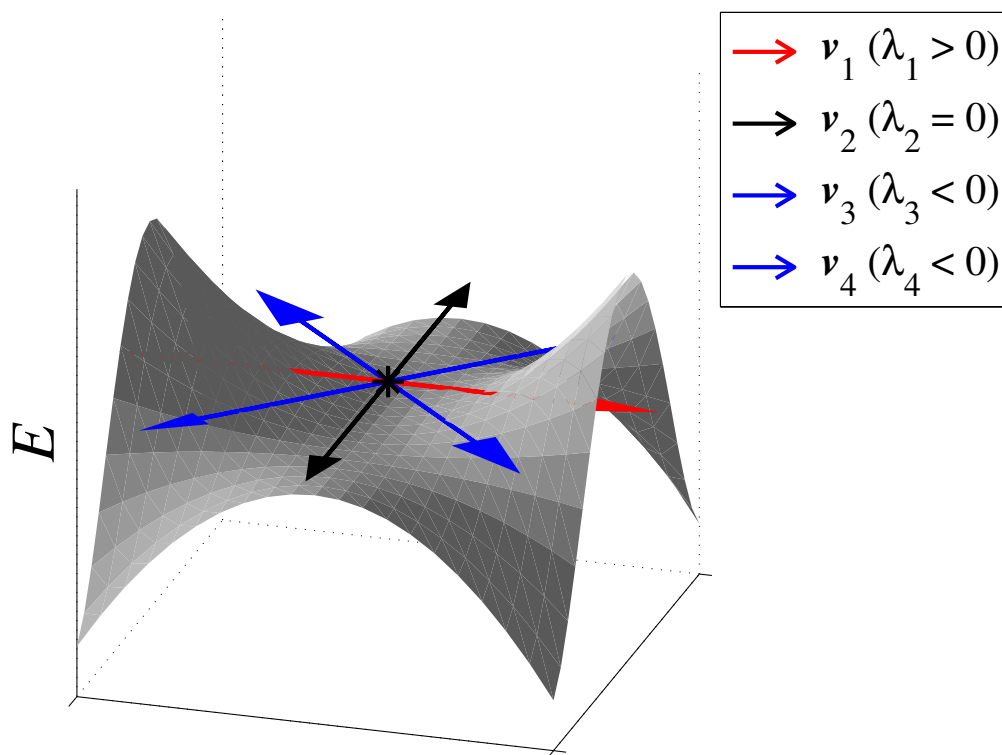


図 4.2: 特異領域上の固有ベクトルの概念図（固有ベクトルは実際には直交している.）

以下ではまず特異領域から探索する方法について説明する.

4.1.1 特異領域からの探索

特異領域は勾配がゼロであるため，従来の勾配を利用した探索法では探索を開始することが出来ない．そこで，最初の重みの更新のときのみ Hesse 行列の固有ベクトルを利用した固有ベクトル降下（EVD: EigenVector Descent）法 [23] を用いることを考える．特異領域上の点の Hesse 行列の固有ベクトルの概念図を図 4.26 に示す．これらの固有ベクトルの中で，負の固有値に対応する固有ベクトルは，特異領域から降下する方向であるため，その方向に重みを更新することで探索を開始できる．以下の計算機実験では固有ベクトル方向への探索幅は 2.2.5 で説明したバックトラッキング法を用いて決定した．

4.1.2 SSF1.2 の処理の概要

SSF1.2 の処理の流れを以下に示す．ただし，考慮する隠れユニットの最大数を $J_{\max}$ とする．

SSF1.2 の処理の流れ

- 1: 隠れユニットが 0 個の実 MLP の大域最適解を求める．($w_0^{(0)} \rightarrow 1/N \sum_{\mu=1}^N y^\mu$)
 - 2: **for** $J = 1, \dots, J_{\max}$ **do**
 - 3: 実 MLP($J - 1$) の探索で得られた最良解に可約性写像を適用して初期点を特異領域上に設定し，Hesse 行列の固有値と固有ベクトルを求める．
 - 4: Step 3 で計算した固有値の中で負のものを選択し，その固有値に対応する固有ベクトル方向とその逆方向に探索空間を降下して解を探索する．
 - 5: **end for**
-

以下の計算機実験では Step 3 で利用する特異領域は $\Theta_J^{\alpha\beta}$ と Θ_J^γ とした．初期点を $\Theta_J^{\alpha\beta}$ 上に設定する方法は以下とした．ただし，実 MLP($J - 1$) の探索で得られた解を $\{w_0^{(J-1)}, w_j^{(J-1)}, \mathbf{w}_j^{(J-1)}, j = 1, \dots, J - 1\}$ ，実 MLP(J) の重みを $\{w_0^{(J)}, w_j^{(J)}, \mathbf{w}_j^{(J)}, j = 1, \dots, J\}$ とする．

$$\begin{aligned} w_0^{(J)} &\rightarrow w_0^{(J-1)}, w_1^{(J)} \rightarrow 0, \mathbf{w}_1^{(J)} \rightarrow (p, 0, \dots, 0)^T, \\ w_j^{(J)} &\rightarrow w_{j-1}^{(J-1)}, \mathbf{w}_j \rightarrow \mathbf{w}_{j-1}^{(J-1)}, j = 2, \dots, J \end{aligned}$$

ここでは $p = -1, 0, 1$ の 3 点とした． Θ_J^γ 上に設定する方法は以下とした．ただし， $m = 2, \dots, J$ とする．

$$\begin{aligned} w_0^{(J)} &\rightarrow w_0^{(J-1)}, w_1^{(J)} \rightarrow q \bullet w_{m-1}^{(J-1)}, w_m^{(J)} \rightarrow (1 - q) \bullet w_{m-1}^{(J-1)}, \\ \mathbf{w}_1^{(J)} &\rightarrow \mathbf{w}_{m-1}^{(J-1)}, \mathbf{w}_m^{(J)} \rightarrow \mathbf{w}_{m-1}^{(J-1)}, w_j^{(J)} \rightarrow w_{j-1}^{(J-1)}, \mathbf{w}_j \rightarrow \mathbf{w}_{j-1}^{(J-1)}, j \in \{2, \dots, J\} \setminus \{m\} \end{aligned}$$

ここでは $q = 0.5, 1.0, 1.5$ （それぞれ内分点，内分と外分の境界点，外分点）の 3 点とした．Step 4 の探索数は選択した特異領域上の点から降下するルートの数に依存する．

4.1.3 計算機実験

提案した SSF1.2 の性能を活性化関数がシグモイド関数の実 MLP を用いて評価した．データとしては人工データ 1 種，実データ 1 種を用いた．比較する従来法は，2.2 で説明した初期重みをランダムに設定した直線探索付き BP と準 Newton 法の一種の BPQ とした．BP と BPQ の探索の初期重みは全て区間 $(-1, +1)$ の中からランダムに選択し，各 J で 100 回の探索を行った．各探索の終了条件は，重み更新回数が 1 万回を超えるか，探索幅が 10^{-16} 以下となった場合とする．プログラミング言語には MATLAB R2014a を用いた．

■ 人工データを用いた実験

人工データは以下の重みを持つ実 MLP を用いて生成した.

$$\begin{aligned}
 (w_0, w_1, \times \times \times, w_{22}) = & (11, 12, 10, 6, 4, 20, 18, 3, 12, 18, 17, 17, 1, \\
 & 13, 9, 9, 3, 14, 1, 18, 15, 12, 13), \\
 (w_1, w_2, \times \times \times, w_{22}) = & \left(\begin{array}{cccccccccccc}
 3 & 7 & 2 & 6 & 8 & 7 & 3 & 10 & 5 & 10 & 3 \\
 3 & 3 & 6 & 2 & 7 & 8 & 9 & 2 & 3 & 1 & 5 \\
 9 & 1 & 4 & 1 & 4 & 3 & 8 & 2 & 4 & 1 & 9 \\
 8 & 8 & 3 & 6 & 7 & 10 & 1 & 5 & 0 & 2 & 4 \\
 3 & 4 & 5 & 7 & 9 & 4 & 4 & 0 & 4 & 6 & 8 \\
 10 & 6 & 1 & 2 & 4 & 5 & 10 & 9 & 5 & 3 & 2 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 1 & 1 & 10 & 6 & 7 & 1 & 1 & 3 & 3 & 7 & 6 \\
 0 & 0 & 8 & 2 & 7 & 4 & 3 & 3 & 7 & 3 & 10 \\
 5 & 10 & 8 & 9 & 7 & 9 & 2 & 2 & 1 & 4 & 1 \\
 5 & 8 & 6 & 1 & 9 & 6 & 3 & 1 & 5 & 2 & 0 \\
 6 & 2 & 2 & 2 & 10 & 4 & 10 & 8 & 9 & 5 & 3 \\
 2 & 2 & 5 & 9 & 8 & 5 & 4 & 3 & 7 & 6 & 5 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{array} \right)
 \end{aligned}$$

訓練データの入力信号は $x_1^\mu, \times \times \times, x_{10}^\mu$ の 10 次元とした. 上記の重みは $x_6^\mu, \times \times \times, x_{10}^\mu$ に対応する重みが全て 0 であるため, $x_6^\mu, \times \times \times, x_{10}^\mu$ は出力に無関係な変数である. x_k^μ の値は領域 $(0, 1)$ からランダムに生成し, 教師信号 y^μ の値は上記の実 MLP の出力値に標準偏差 0.05 の正規乱数を加え, 1000 個のサンプル点を生成した. テスト誤差は訓練データとは別に生成した 1,000 個のノイズなしサンプル点により計算した. 隠れユニットの最大数は 24 とした.

図 4.3, 4.4 に各 J での最小の訓練誤差とそのときのテスト誤差を示す. BP の訓練誤差は, $J = 11$ 以降, 他の探索法の訓練誤差より大きい結果となった. BPQ の訓練誤差は

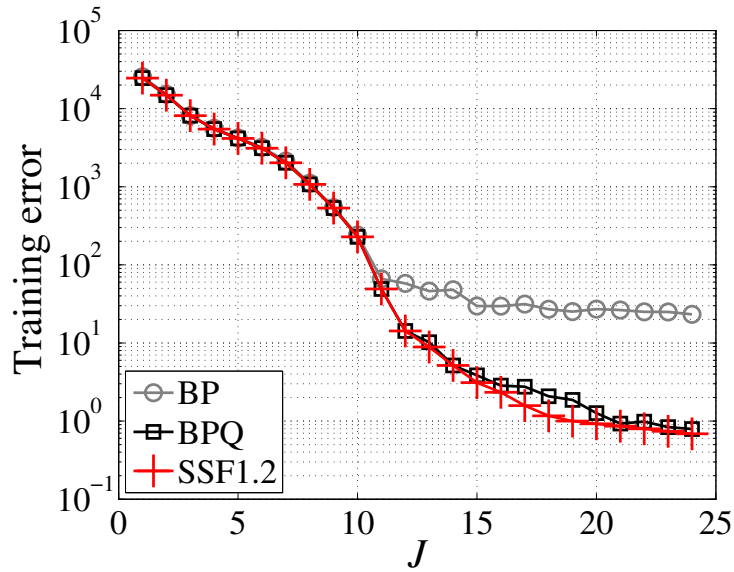


図 4.3: BP, BPQ, SSF1.2 の訓練誤差 (人工データ)

表 4.1: BP, BPQ, SSF1.2 の合計処理時間 (人工データ)

	BP	BPQ	SSF1.2
合計処理時間 (hr:min:sec)	04:02:19	05:33:31	11:19:07

$J = 15$ 以降 SSF1.2 よりも大きく、テスト誤差は $J = 22$ のとき最小となったが、SSF1.2 に比べて不安定な動きを示した。これは、BPQ では初期値を乱数で決めるため、訓練誤差の値が近かったとしても重みの値が大きく異なることがあることが原因と考えられる。逆に SSF1.2 のテスト誤差は安定していることがわかる。SSF1.2 のテスト誤差は $J = 18$ のとき最小となり、BPQ により得た最小テスト誤差よりも小さいテスト誤差の解を得た。データを生成した実 MLP は $J = 22$ であったが、SSF1.2 のテスト誤差は $J = 22$ のとき最小とならなかった。これは幾つかの隠れユニットが冗長な働きをしていることが原因と考えられる。

図 4.5, 4.6 に各 J での処理時間と重み更新平均回数を、表 4.1 に合計処理時間を、図 4.7 に SSF1.2 の探索数を示す。BP の重み更新平均回数は常に 1 万回となった。BPQ は隠れユニットの増加に伴って重み更新平均回数が増加する傾向があったが、SSF1.2 は重み更新平均回数が $J = 3$ 以外は 4 千回以下となった。これは SSF1.2 では実 MLP($J - 1$) の解に可約性写像を適用して形成された特異領域から探索するため、探索の開始点が実 MLP(J) の解に近いと考えられる。しかし、SSF1.2 は最も時間が掛かった。これは SSF1.2 の

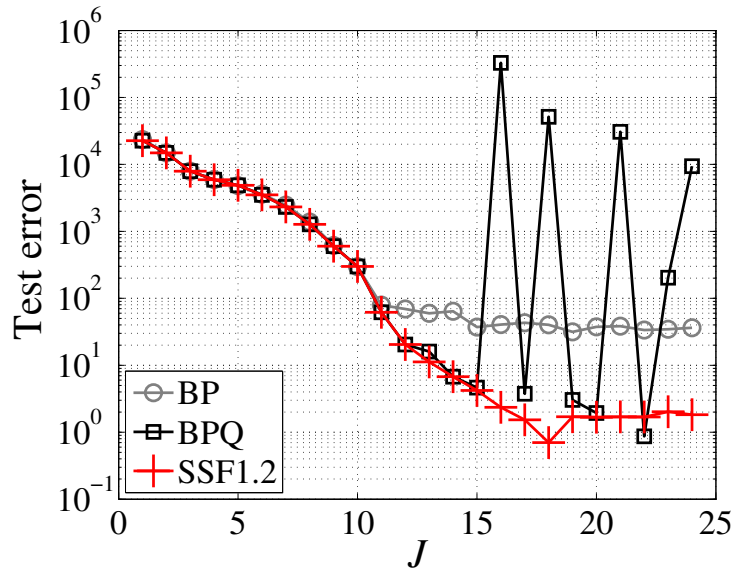


図 4.4: BP, BPQ, SSF1.2 のテスト誤差 (人工データ)

探索数が隠れユニットの増加に伴って増加する傾向があったためと考えられる。

■ 実データを用いた実験

実データとしては UCI ML Repository より入手 Parkinson telemonitoring データを用いた [12]。サンプル数は 5875 あり，発声測定値などの 18 入力変数から，パーキンソン病の階級 (motor UPDRS) を推定する。訓練データとしては全データを 10 分割し，その内 9 セグメントを用い，残りの 1 セグメントをテストデータとして用いた。隠れユニットの最大数は 9 とした。

図 4.8, 4.9 に各 J での各探索法の最小の訓練誤差とそのときのテスト誤差を示す。BP では隠れユニットを増加してもほとんど訓練誤差が減少しなかった。BPQ と SSF1.2 の訓練誤差とテスト誤差は隠れユニットの増加に伴って単調減少したが， $J = 8, 9$ のときは SSF1.2 が最も小さい訓練誤差とテスト誤差の解を得た。

図 4.10, 4.11 に各 J での処理時間と重み更新平均回数を，表 4.2 に合計処理時間を，図 4.12 に SSF1.2 の探索数を示す。BP の重み更新平均回数は，人工データを用いた実験と同様，常に 1 万回となった。BPQ と SSF1.2 の重み更新平均回数は人工データの実験と比べて多い傾向があった。処理時間は BP と BPQ でも隠れユニットの増加に伴って増加したが，SSF1.2 ではより大幅に処理時間が増加した。これは，人工データを用いた実験と

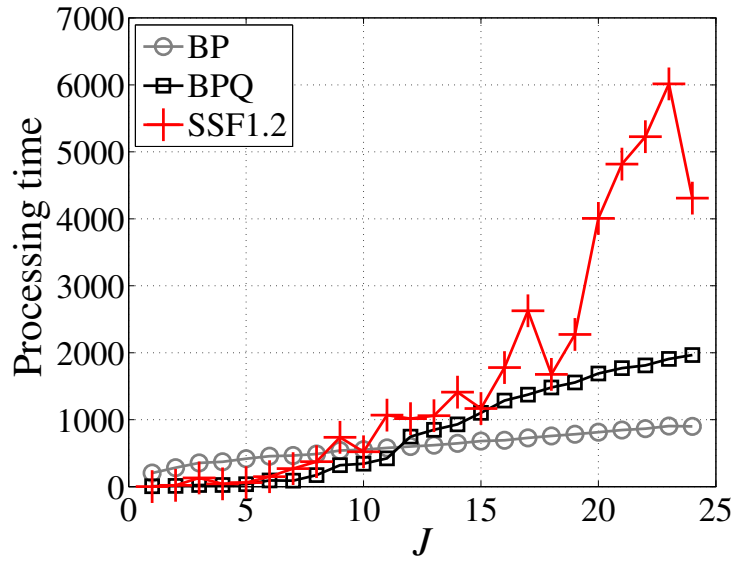


図 4.5: BP, BPQ, SSF1.2 の処理時間 (秒) (人工データ)

表 4.2: BP, BPQ, SSF1.2 の合計処理時間 (実データ)

	BP	BPQ	SSF1.2
合計処理時間 (hr:min:sec)	02:39:43	03:19:00	17:14:58

同様、SSF1.2 の探索数が隠れユニットが増加に伴って増加する傾向があったためと考えられる。

4.1.4 考察

4.1では実MLP($J-1$)の最適解に可約性写像を適用して形成した特異領域から実MLP(J)の探索を開始するSSF1.2を提案した。初期重みをランダムに設定しBPやBPQを用いて解を探索する方法では隠れユニットを増やしても訓練誤差が減少しないことがあったが、SSF1.2は隠れユニットの増加に伴って訓練誤差を単調減少させ、安定して良質の解を得ることができた。しかし、隠れユニットの増加に伴って処理時間が増大する傾向があった。

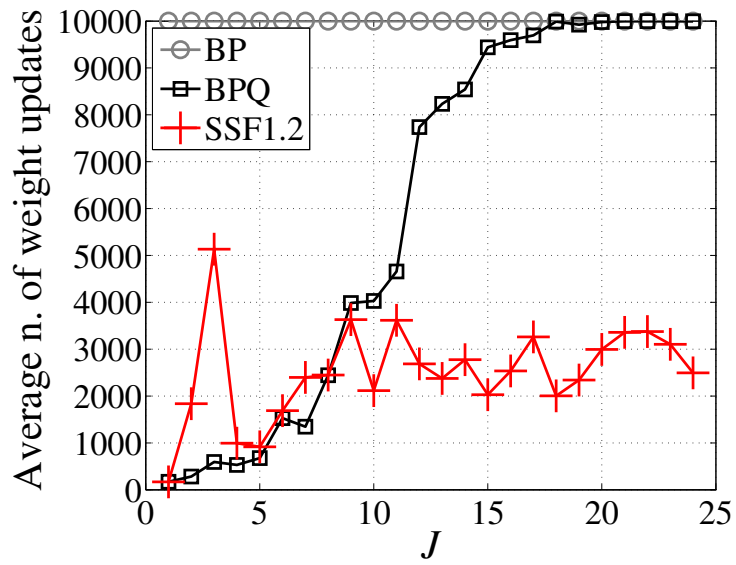


図 4.6: BP, BPQ, SSF1.2 の重み更新平均回数 (人工データ)

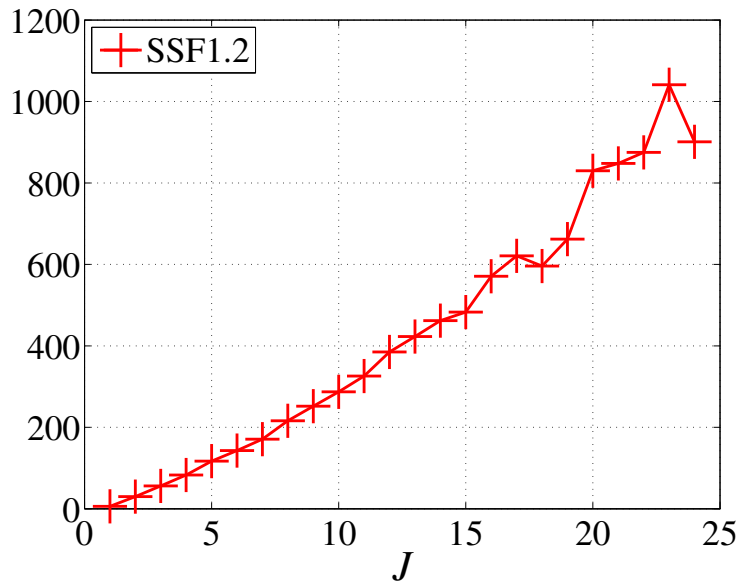


図 4.7: SSF1.2 の探索数 (人工データ)

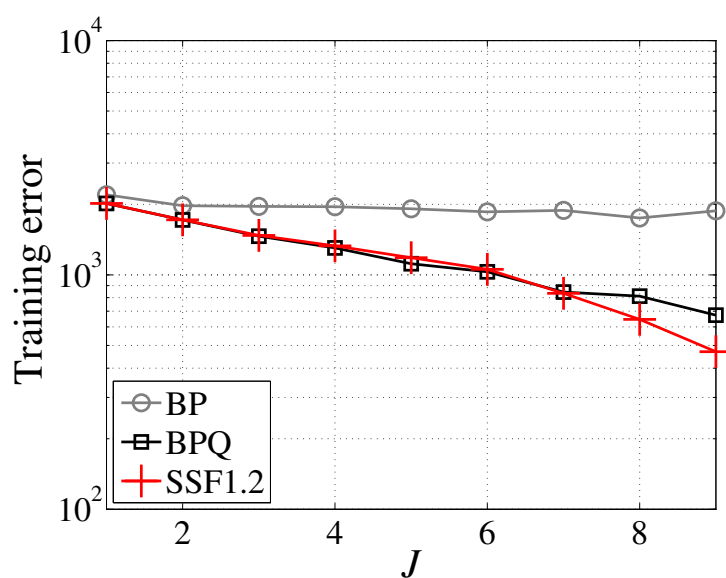


図 4.8: BP, BPQ, SSF1.2 の訓練誤差 (実データ)

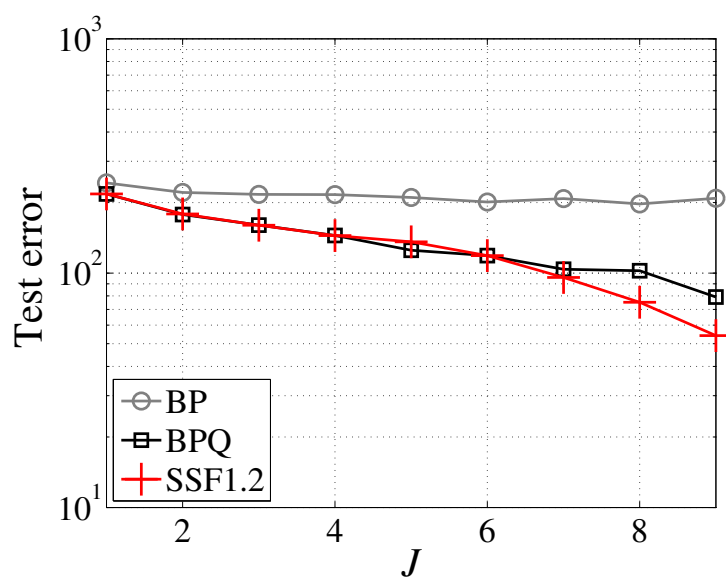


図 4.9: BP, BPQ, SSF1.2 のテスト誤差 (実データ)

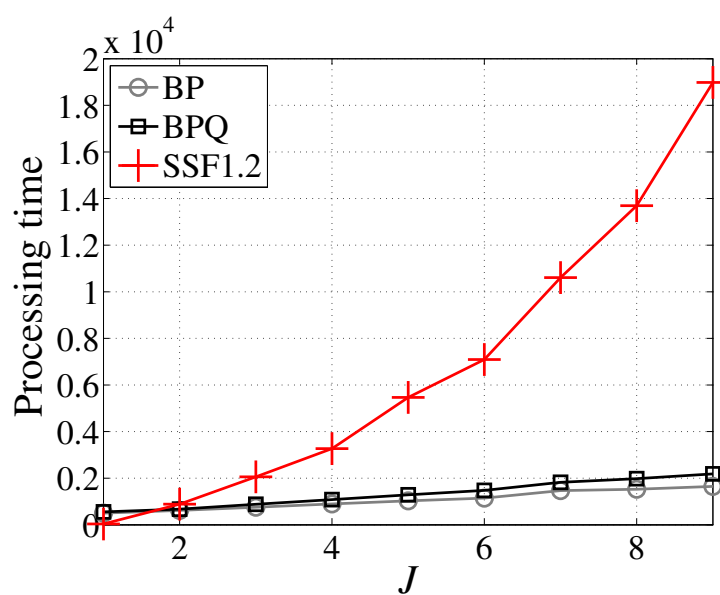


図 4.10: BP, BPQ, SSF1.2 の処理時間 (秒) (実データ)

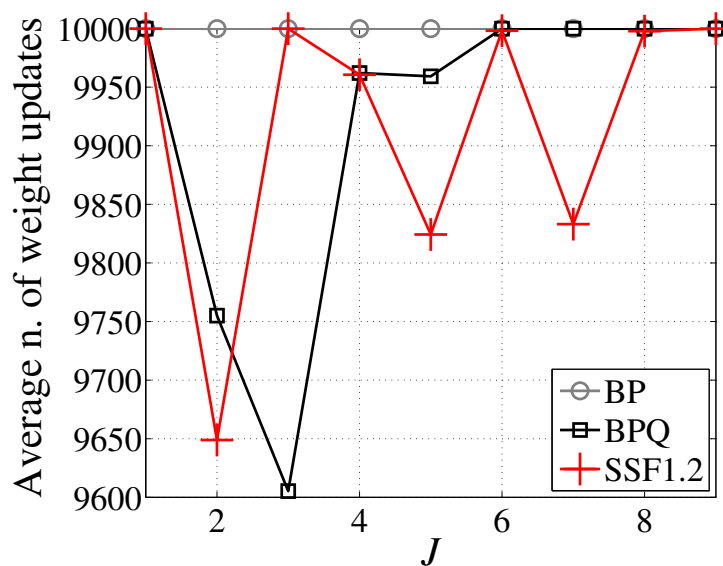


図 4.11: BP, BPQ, SSF1.2 の重み更新平均回数 (実データ)

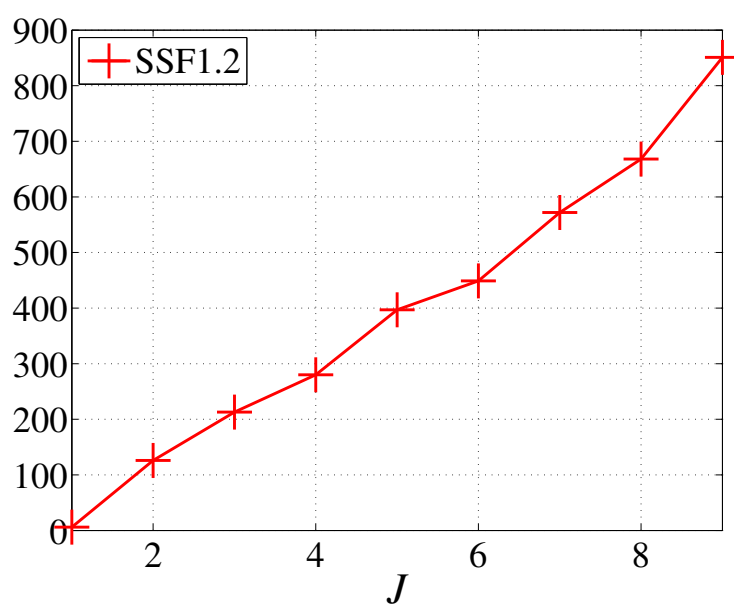


図 4.12: SSF1.2 の探索数 (実データ)

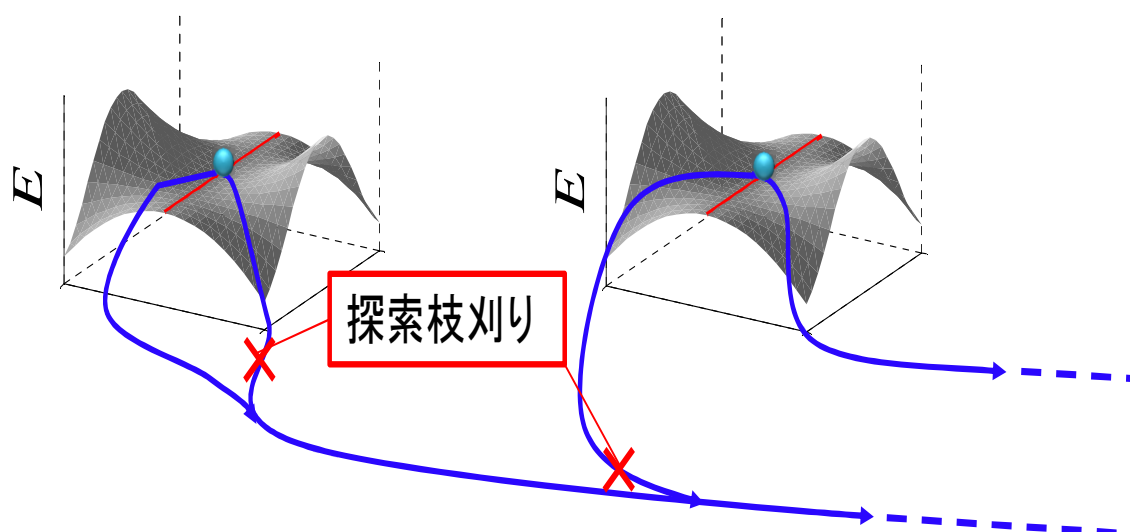


図 4.13: 探索枝刈りの概念図

4.2 SSF1.3

SSF1.2 は実 $\text{MLP}(J-1)$ の最適解に可約性写像を適用して形成された特異領域から探索空間を降下するため、訓練誤差を単調に減少させられる。しかし、隠れユニットの増加とともに探索空間の次元が増え、特異領域も増えるため、全ての特異領域から探索すると処理負荷が増大してしまう。そこで、現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を SSF1.2 に導入した SSF1.3 を提案する。

4.2.1 探索枝刈りによる高速化

探索枝刈り処理では、図 4.13 に示すように、現在の探索が以前と同じ探索経路に合流したかどうかを判定し、合流したと判定した場合はその探索を枝刈りする。SSF1.2 は特異領域上の複数の点から複数方向へ探索を行うため、探索枝刈りにより、処理時間の大幅な削減が期待できる。

■ 探索枝刈り判定

いま、現在探索中の点を $\theta^{(t)}$ 、過去の探索の通過点を $\phi^{(i)}$ 、正規化係数ベクトルを $\mathbf{d}$ とする。ここでは、絶対値が大きい重みのみで判定されることを防ぐため、以下のように正規化する。

$$d_m \rightarrow \frac{1}{\theta_m^{(t-1)}} \begin{cases} (1 < \theta_m^{(t-1)}) \\ (\theta_m^{(t-1)} \geq 1) \end{cases} \quad (4.1)$$

$$\mathbf{v}^{(t)} \rightarrow \text{diag}(\mathbf{d}) \theta^{(t)} \quad (4.2)$$

$$\mathbf{v}^{(t-1)} \rightarrow \text{diag}(\mathbf{d}) \theta^{(t-1)} \quad (4.3)$$

$$\mathbf{r}^{(i)} \rightarrow \text{diag}(\mathbf{d}) \phi^{(i)}, i = 1, \dots, I \quad (4.4)$$

ただし、 $m = 1, 2, \dots, M$ 、 M は重みの数、 I は過去の探索中に記録した点の数とする。 $\text{diag}(\mathbf{d})$ は $\mathbf{d}$ を対角要素とし、他は0の行列とする。ここでは図 4.14 に示すように、 $\mathbf{v}^{(t-1)}$ と $\mathbf{v}^{(t)}$ を結んだ線分を含む直線と $\mathbf{r}^{(i-1)}$ と $\mathbf{r}^{(i)}$ を結んだ線分を含む直線の両方に直交する最短の線分を $\mathbf{l}$ とし、 $\mathbf{l}$ を基に過去の探索と同じルートを通っているか判定する。同じルートを通っていると判定した場合は探索を枝刈りする。

ここで $\mathbf{l}$ を求めることを考える。二つの線分の始点をそれぞれ $\mathbf{v}^{(t-1)}$ 、 $\mathbf{r}^{(i-1)}$ 、終点をそれぞれ $\mathbf{v}^{(t)}$ 、 $\mathbf{r}^{(i)}$ とすると、二つの線分を含む直線間の最短の線分は以下で表すことができる。ただし、 $\Delta \mathbf{r}^{(i)} \leq \mathbf{r}^{(i)} - \mathbf{r}^{(i-1)}$ 、 $\Delta \mathbf{v}^{(t)} \leq \mathbf{v}^{(t)} - \mathbf{v}^{(t-1)}$ とする。

$$\mathbf{l} = \mathbf{r}^{(i-1)} + a_1 \Delta \mathbf{r}^{(i)} - (\mathbf{v}^{(t-1)} + a_2 \Delta \mathbf{v}^{(t)}) \quad (4.5)$$

上記式の a_1 と a_2 は未知の変数であるが、以下の式を満たすとき $\mathbf{l}$ の始点と終点が二つの線分上に存在する。

$$0 \leq a_1 \leq 1, \quad 0 \leq a_2 \leq 1 \quad (4.6)$$

$\mathbf{a} = (a_1, a_2)^T$ は以下の最小化問題を解けば得られる。

$$\min_{\mathbf{a}} \mathbf{l}^T \mathbf{l} \quad (4.7)$$

$\mathbf{a}$ は以下のように求まる。ただし、 $b_1 \leq \|\Delta \mathbf{r}^{(i)}\|^2$ 、 $b_2 \leq \Delta \mathbf{r}^{(i)T} \Delta \mathbf{v}^{(t)}$ 、 $b_3 \leq \|\Delta \mathbf{v}^{(t)}\|^2$ 、 $b_4 \leq (\mathbf{r}^{(i-1)} - \mathbf{v}^{(t-1)})^T \Delta \mathbf{r}^{(i)}$ 、 $b_5 \leq (\mathbf{r}^{(i-1)} - \mathbf{v}^{(t-1)})^T \Delta \mathbf{v}^{(t)}$ とする。

$$\mathbf{a} \rightarrow \begin{pmatrix} 1 & b_3 b_4 & b_2 b_5 \\ b_1 b_3 & b_2^2 & b_2 b_5 \\ b_2 b_4 & b_1 b_5 & b_3 b_5 \end{pmatrix}^{-1} \begin{pmatrix} b_3 b_4 & b_2 b_5 \\ b_2 b_4 & b_1 b_5 \end{pmatrix} \quad (4.8)$$

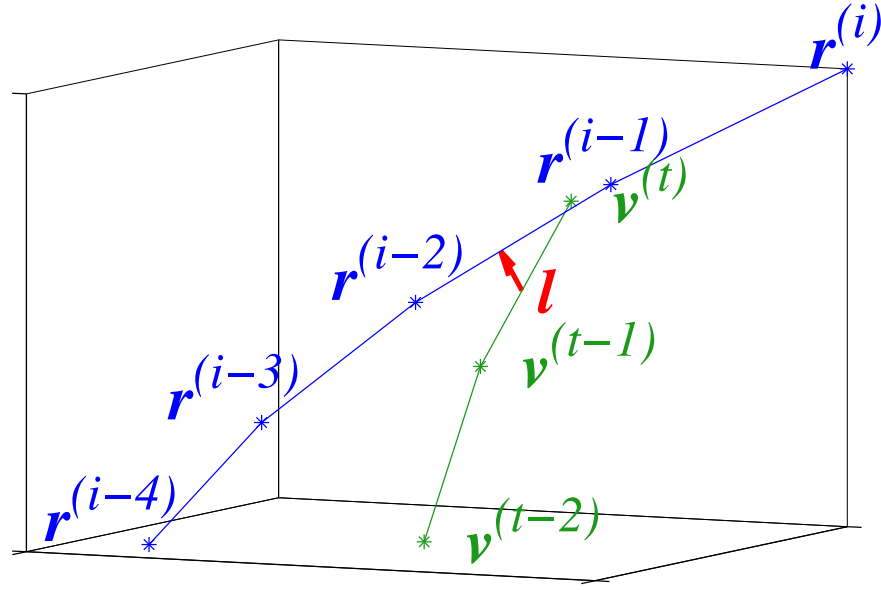


図 4.14: 探索枝刈りの概念図（軸は全て重み）

以下の計算機実験では、式 (4.6) を満たさない場合は以前と同じルートを通っていないと判定した。満たす場合は式 (4.5) により l を求め、全ての要素が $l_m < \epsilon$ を満たすとき、探索を枝刈る。 ϵ は 0.3 とし、この判定と θ, ϕ の記録は重み更新回数が 100 の倍数のときとした。

なお、判定を l のノルムで判定する方法も考えられるが、その場合は閾値を $M\epsilon$ のようにして重みの増加に伴って閾値を変える必要がある。しかし、不要変数が複数ある場合は、不要変数を取り除いた場合と比べると重みの数が多いため、探索が枝刈りされやすくなることも考慮しなければならない。一方、 l の各要素で判定すれば、不要変数の存在に影響されない。

■ 探索枝刈り判定の前処理

本稿の実験では実 MLP の活性化関数としてシグモイド関数を用いるが、シグモイド関数 $\sigma(x)$ は、図 4.15 に示すように、点 $(x = 0, \sigma(x) = 1/2)$ を中心に点対称であり、以下の

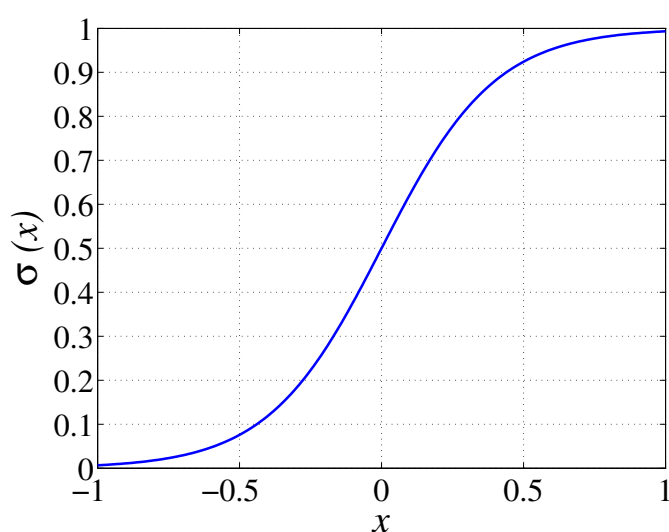


図 4.15: シグモイド関数

式が成り立つ.

$$\sigma(x) = 1 - \sigma(-x) \quad (4.9)$$

そのため, ある隠れユニット j から出力ユニットへ入力される信号は, 以下の式が成り立つ.

$$w_j \sigma(\mathbf{w}_j^T \mathbf{x}) = |w_j| \sigma(\text{sgn}(w_j) \mathbf{w}_j^T \mathbf{x}) \quad (4.10)$$

この性質のため, 以下のように重みを入れ替えても実 MLP の機能は変わらない.

$$w_0 \rightarrow w_0 - w_j, \quad w_j \rightarrow |w_j|, \quad \mathbf{w}_j \rightarrow \text{sgn}(w_j) \mathbf{w}_j \quad (4.11)$$

そのため, 探索枝刈り判定を行う前に w_j の符号を正にそろえ, 上記のシグモイド関数の性質を考慮した探索の枝刈りを行う. その他の奇関数や偶関数を活性化関数として用いる場合も関数の性質を考慮した探索の枝刈りが可能である.

また, 隠れユニットの並びは, 活性化関数の種類に依らず, 順序を入れ替えても実 MLP の機能は変わらない. そのため, w_j の符号をそろえた後, w_j が昇順となるように隠れユニットの順番を入れ替える前処理を行う. この処理により, 隠れユニットの並びを考慮した探索の枝刈りが可能となる.

ただし, これらの重みの変換は探索枝刈り判定のためにのみ行う. そのため, 変換後の重みを次の探索の判定のために記録するが, 実際の探索では変換後の重みを反映させない.

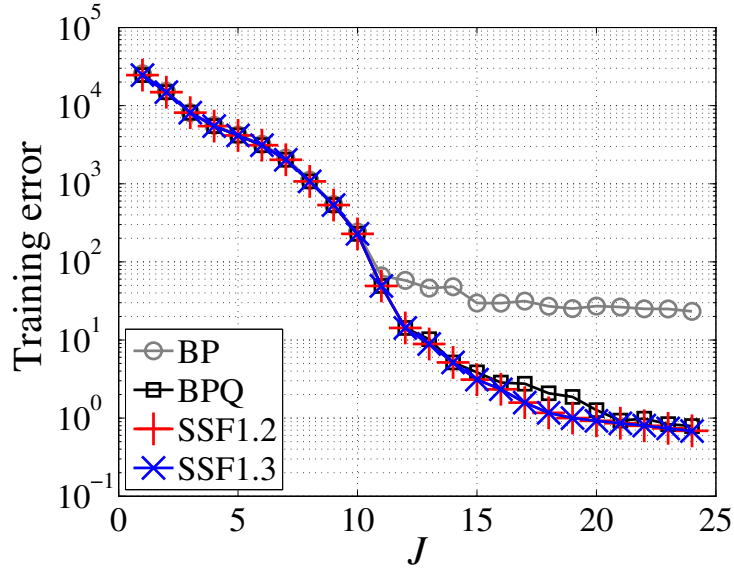


図 4.16: SSF1.3 の訓練誤差 (人工データ)

表 4.3: SSF1.3 の合計処理時間 (人工データ)

	BP	BPQ	SSF1.2	SSF1.3
合計処理時間 (hr:min:sec)	04:02:19	05:33:31	11:19:07	04:04:57

4.2.2 計算機実験

提案した SSF1.3 の性能を 4.1.3 と同様のデータを用いて評価した。

■ 人工データを用いた実験

4.1.3 と同じ人工データを用いて実験を行った。図 4.16, 4.17 に各 J での最小の訓練誤差とそのときのテスト誤差を示す。SSF1.3 では、SSF1.2 と同様、訓練誤差が単調減少し、 $J = 18$ のとき最小のテスト誤差の解を得た。

図 4.18, 4.19 に各 J での処理時間と重み更新平均回数を、表 4.3 に合計処理時間を、図 4.20 に SSF1.3 の探索開始数と最後まで探索枝刈りされなかった探索数を示す。ただし、SSF1.3 の重み更新平均回数は探索枝刈りされなかった探索の重み更新回数の平均とした。SSF1.3 の重み更新平均回数は SSF1.2 より減少した。また、SSF1.3 の探索開始数は SSF1.2 とほとんど同じであるが、SSF1.3 では 50 % (5862 / 11661) の探索を枝刈りすることにより、処理時間を削減し、SSF1.2 よりも 2 倍以上速く処理が終了し、BPQ よりも速く処理

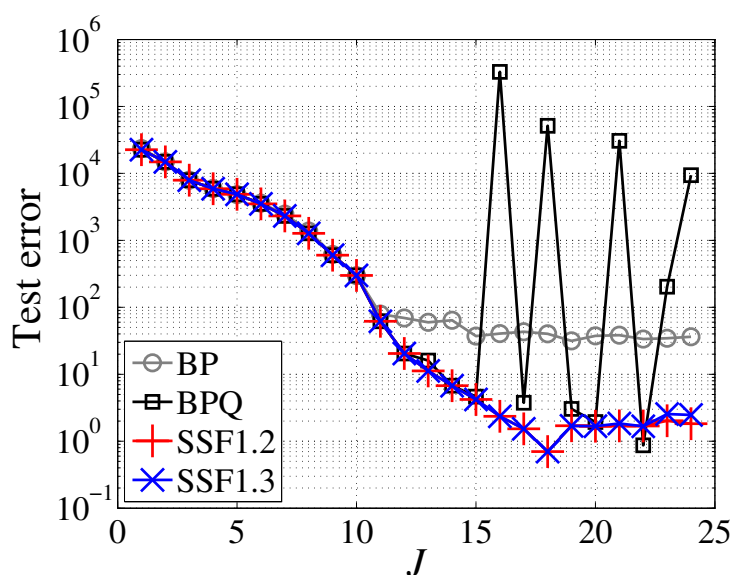


図 4.17: SSF1.3 のテスト誤差 (人工データ)

が終了した.

■ 実データを用いた実験

4.1.3 と同じ実データを用いて実験を行った. ただし, 隠れユニットの最大数は 18 に変更した. SSF1.2 は $J = 9$ のとき 5 時間以上処理時間が掛かったため $J = 9$ で処理を打ち切った.

図 4.21, 4.22 に各 J での各探索法の最小の訓練誤差とそのときのテスト誤差を示す. BP は隠れユニットを 18 まで増やしてもほとんど訓練誤差が減少しなかった. BPQ は減少傾向にあるが, $J = 14, 16$ のとき, $J - 1$ のときの訓練誤差よりも訓練誤差が増加した. SSF1.3 は, SSF1.2 と同様, 訓練誤差を単調減少させ, BPQ により得たテスト誤差よりも小さいテスト誤差の解を得た.

図 4.23, 4.24 に各 J での処理時間と重み更新平均回数を, 表 4.4 に合計処理時間を, 図 4.25 に SSF1.3 の探索数を示す. ただし, SSF1.3 の重み更新平均回数は探索枝刈りされなかった探索の重み更新回数の平均とした. 各探索法の重み更新平均回数はほとんど全て 1 万回となった. SSF1.3 の重み更新平均回数は $J = 18$ のとき約 5000 回, $J = 16, 17$ のときは 1000 回以下となり, かつ SSF1.3 では 76 % (14360 / 18949) の探索を枝刈りしたが, BPQ よりも処理時間が掛かった. これは SSF1.3 では探索開始数が増加傾向にあり, 探索

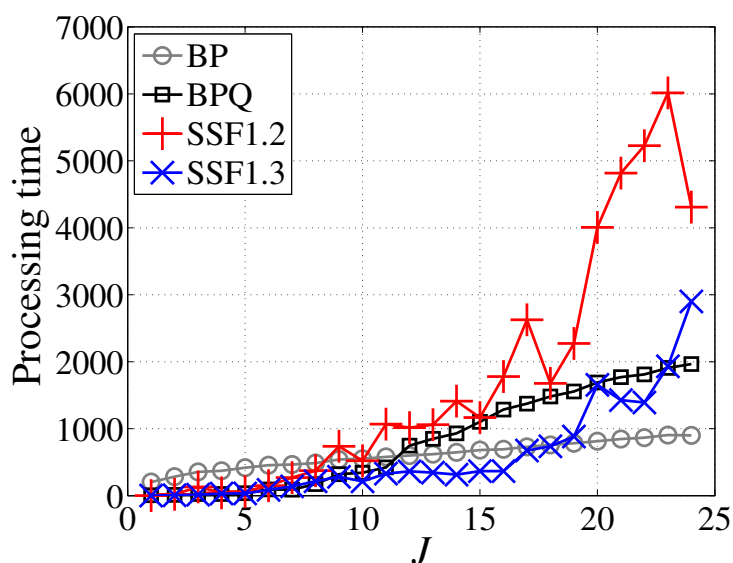


図 4.18: SSF1.3 の処理時間 (秒) (人工データ)

表 4.4: SSF1.3 の合計処理時間 (実データ)

	BP	BPQ	SSF1.2	SSF1.3
合計処理時間 (hr:min:sec)	07:49:48	12:07:49	-	25:02:43

枝刈りをしたとしても、探索枝刈り判定されるまでに少なくとも 200 回は重みを更新しなければならないため、時間が掛かったと考えられる。

4.2.3 考察

SSF1.3 は、現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を SSF1.2 に導入することにより、解品質を劣化させることなく高速化できた。人工データを用いた実験では SSF1.2 と同じように最小のテスト誤差を $J = 18$ のときに得るとともに、処理時間も高速化され、BPQ や SSF1.2 よりも速く処理が終了した。実データを用いた実験でも BPQ よりも良質の解を得、SSF1.2 よりも高速化されたが、以前として隠れユニット数の増加に伴って処理時間が増え、BPQ よりも時間が掛かった。

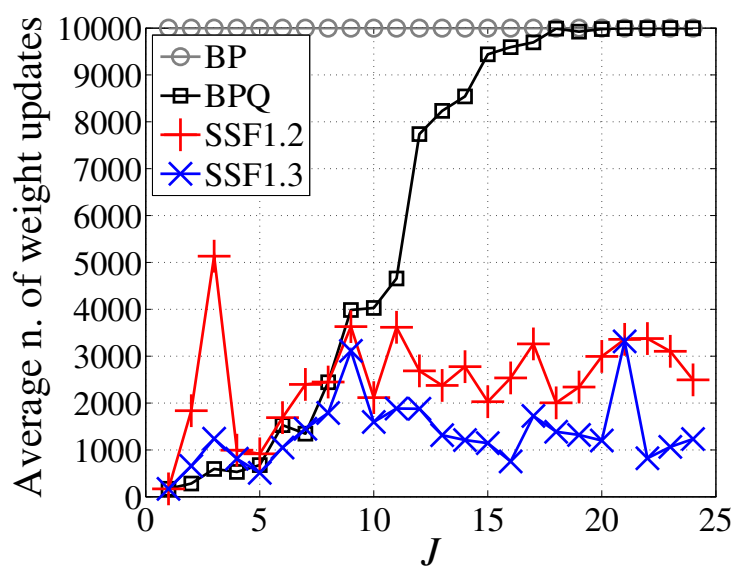


図 4.19: SSF1.3 の重み更新平均回数 (人工データ)

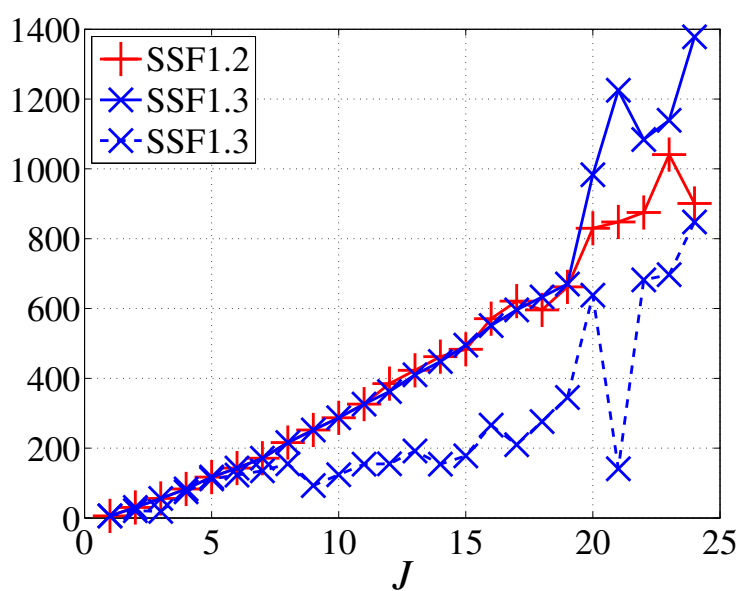


図 4.20: SSF1.3 の探索開始数 (実線) と最後まで探索枝刈りされなかった探索数 (点線) (人工データ)

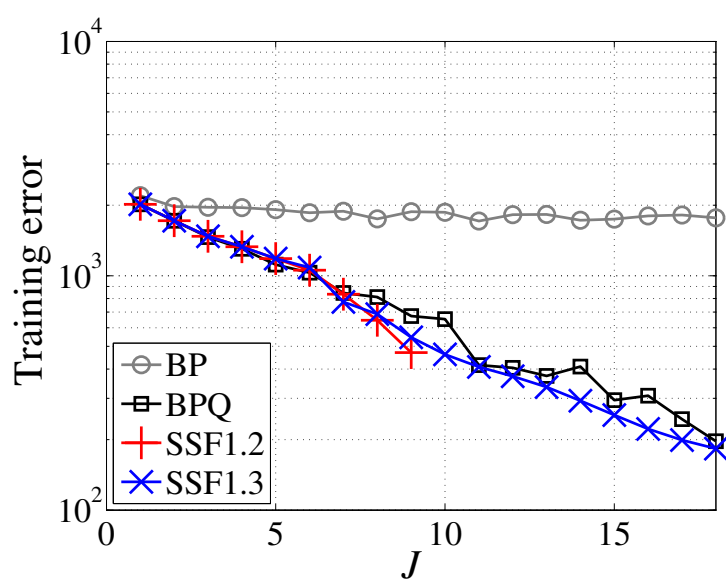


図 4.21: SSF1.3 の訓練誤差 (実データ)

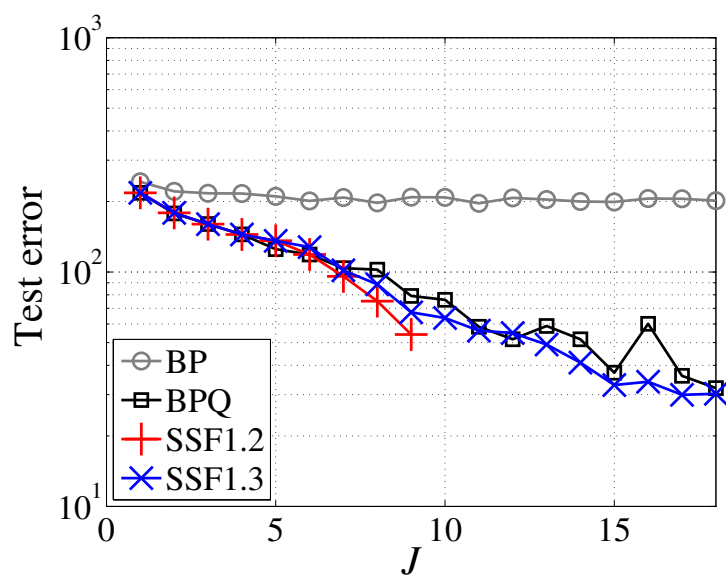


図 4.22: SSF1.3 のテスト誤差 (実データ)

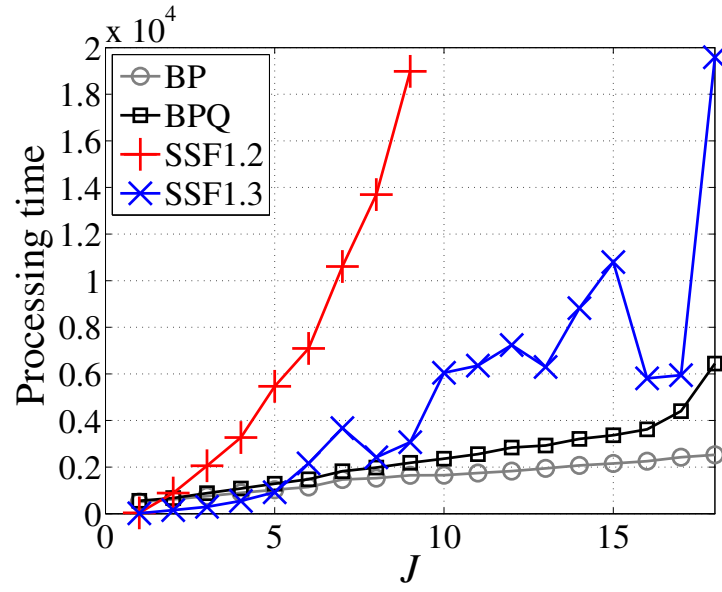


図 4.23: SSF1.3 の処理時間 (秒) (実データ)

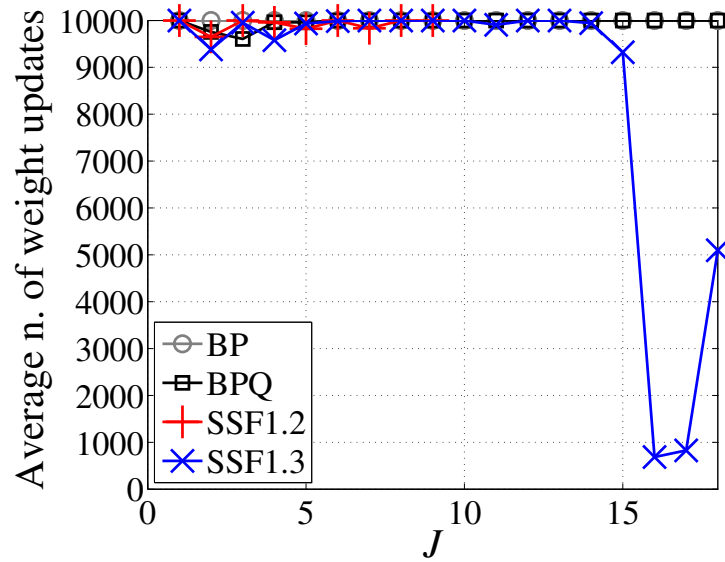


図 4.24: SSF1.3 の重み更新平均回数 (実データ)

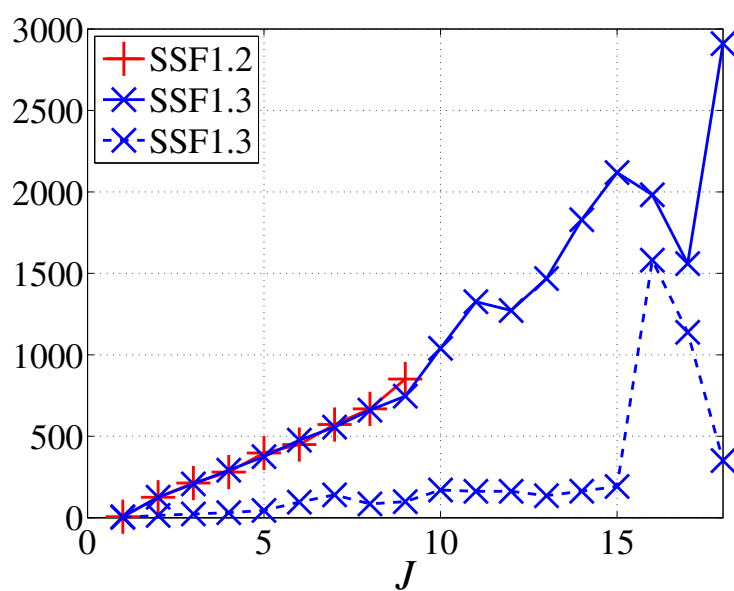


図 4.25: SSF1.3 の探索数 (実データ)

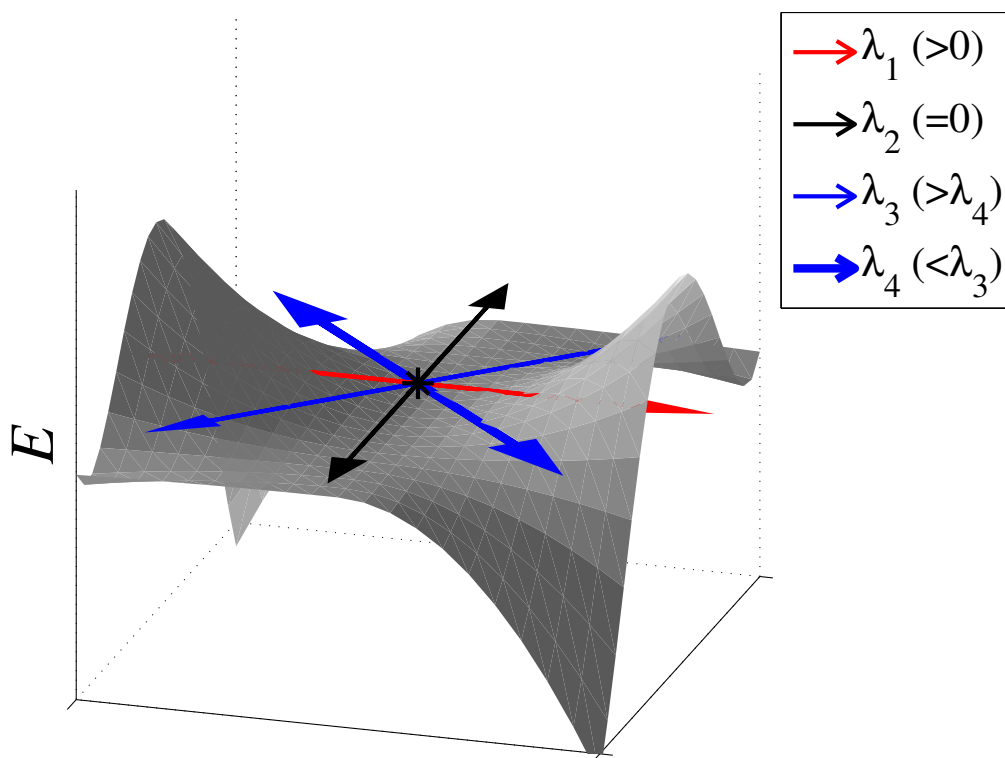


図 4.26: 複数の負の固有値を持つ特異領域の概念図（固有ベクトルは実際には直交している。）

4.3 SSF1.4

SSF1.3はSSF1.2に探索枝刈りを導入することにより高速化されたが、隠れユニットが多いとき従来法よりも時間が掛かる傾向があった。そこで、4.3ではSSF1.3を更に高速化したSSF1.4を提案する。SSF1.4では、各隠れユニット毎に実行する探索数に上限を設定し、特異領域上のHesse行列の固有値が小さい順に探索を行うことで高速化する。このことにより、解品質を劣化させることなく高速化することを目指す。

4.3.1 探索本数削減による高速化

図 4.26 に示すように探索空間を局所的に見ると、負の固有値の内、固有値が小さいほどその固有値に対応する固有ベクトル方向の探索空間のたわみはより急である。そのため、負の固有値が小さいほど、対応する固有ベクトル方向の訓練誤差は局所的に見ると小

さい。しかし、最小の固有値に対応する固有ベクトル方向へ探索すると必ず最小の訓練誤差の解が得られる訳ではないため、SSF1.4 では小さい固有値から優先的に選択して複数回の探索を行う。

4.3.2 SSF1.4 の処理の概要

SSF1.4 の処理の流れを以下に示す。ただし、考慮する隠れユニットの最大数を $J_{\max}$ とする。

SSF1.4 の処理の流れ

- 1: 隠れユニットが 0 個の実 MLP の大域最適解を求める。 ($w_0^{(0)} \rightarrow 1/N \sum_{\mu=1}^N y^\mu$)
 - 2: **for** $J = 1, \dots, J_{\max}$ **do**
 - 3: 実 MLP($J - 1$) の探索で得られた最良解に可約性写像を適用して初期点を特異領域上に設定し、Hesse 行列の固有値と固有ベクトルを求める。
 - 4: **while** 終了条件を満たすまで以下を繰り返す。 **do**
 - 5: Step 3 で計算した負の固有値の中で最も小さい固有値から順に選択し、その固有値に対応する固有ベクトル方向とその逆方向に探索空間を降下して解を探索する。
 - 6: **end while**
 - 7: **end for**
-

以下の計算機実験では Step 3 で利用する特異領域は SSF1.2 と同じとした。Step 5 の終了条件は探索枝刈りされた探索も含めて探索数が 100 となるか、選択する負の固有値が無くなったときとした。

4.3.3 計算機実験

提案した SSF1.4 の性能を 4.1.3 と同様のデータを用いて評価した。

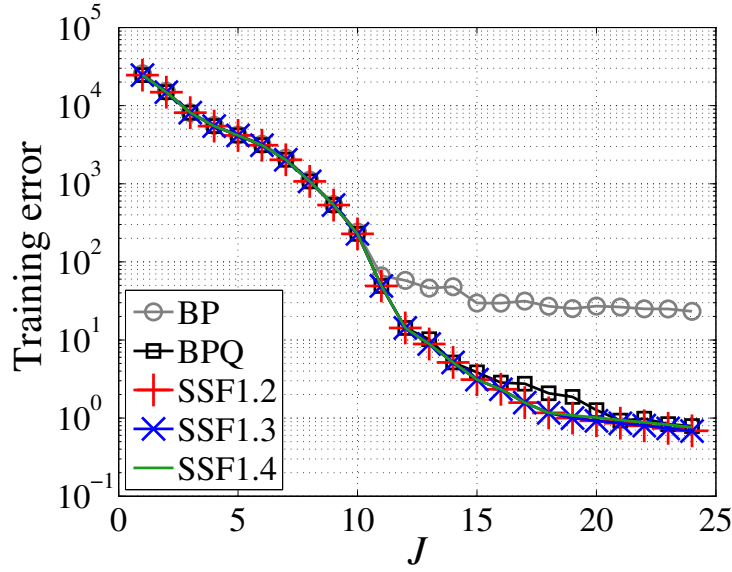


図 4.27: SSF1.4 の訓練誤差 (人工データ)

表 4.5: SSF1.4 の合計処理時間 (人工データ)

	BP	BPQ	SSF1.2	SSF1.3	SSF1.4
合計処理時間 (hr:min:sec)	04:02:19	05:33:31	11:19:07	04:04:57	01:15:59

■ 人工データを用いた実験

4.1.3 と同じ人工データを用いて実験を行った．図 4.27, 4.28 に各 J での最小の訓練誤差とそのときのテスト誤差を示す．SSF1.4 では，SSF1.2, SSF1.3 と同様，訓練誤差が単調減少し， $J = 18$ のとき最小のテスト誤差の解を得た．

図 4.29, 4.30 に各 J での処理時間と重み更新平均回数を，表 4.5 に合計処理時間を，図 4.31 に SSF1.4 の探索開始数と最後まで探索枝刈りされなかった探索数を示す．ただし，SSF1.4 の重み更新平均回数は，SSF1.3 と同様，探索枝刈りされなかった探索の重み更新回数の平均とした．SSF1.4 の探索開始数は $J = 5$ 以降全て上限の 100 に達したが，解品質は SSF1.2, 1.3 と同様良質の解を得た．また，SSF1.4 は探索開始数を 100 回に制限することで合計処理時間が最短となった．SSF1.4 の処理速度は SSF1.3 よりも 3 倍以上であった． $J = 5$ 以降の SSF1.4 の探索開始数は，BP や BPQ と同様，100 であったにも関わらず，隠れユニット数が 1 から 24 まで全て SSF1.4 の方が 4 倍以上 BPQ よりも速かった．これは SSF1.4 では探索枝刈りにより高速化されていることに加え， $J = 5$ 以降の SSF1.4 の重み更新平均回数が BPQ よりも少ないことが原因と考えられる．

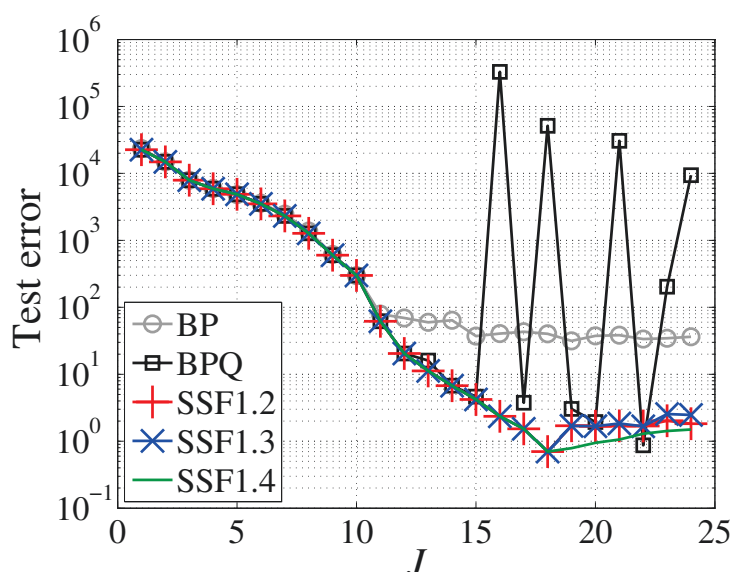


図 4.28: SSF1.4 のテスト誤差 (人工データ)

■ 実データを用いた実験

4.1.3 と同じ実データを用いて実験を行った。ただし、隠れユニットの最大数は 32 に変更して BPQ と SSF1.4 を実行した。SSF1.3 は $J = 18$ のとき 5 時間以上処理時間が掛かったため $J = 18$ で処理を打ち切った。また、BP の解品質は隠れユニットを増やしても改善の傾向が見られないため SSF1.3 と同様 $J = 18$ で処理を打ち切った。

図 4.32, 4.33 に各 J での最小の訓練誤差とそのときのテスト誤差を示す。BPQ では $J = 18$ 以下のとき $J - 1$ よりも訓練誤差が増加した回数は $2/17(12\%)$ であったが、19 以降では $8/14(57\%)$ であり、隠れユニットが増えるほど訓練誤差が単調減少しづらくなった。これは隠れユニットが増えるほど局所最適解や特異領域が増えることで、訓練誤差が比較的大きい局所最適解や特異領域に収束しやすくなったためと考えられる。一方、SSF1.4 の訓練誤差は隠れユニットの増加に伴って単調減少し、 $J = 18$ 以降では BPQ との差がより一層大きくなった。BPQ と SSF1.4 のテスト誤差はそれぞれ $J = 26, 29$ のとき最小となり、SSF1.4 の方が小さいテスト誤差の解を得た。

図 4.34, 4.35 に各 J での処理時間と重み更新平均回数を、表 4.6 に合計処理時間を、図 4.36 に SSF1.3 の探索数を示す。ただし、SSF1.4 の重み更新平均回数は、SSF1.3 と同様、探索枝刈りされなかった探索の重み更新回数の平均とした。SSF1.4 の重み更新平均回数は $J = 2, \infty, 5$ 以外は全て 1 万回となり、BPQ の重み更新平均回数も同じように $J = 2, \infty, 5$

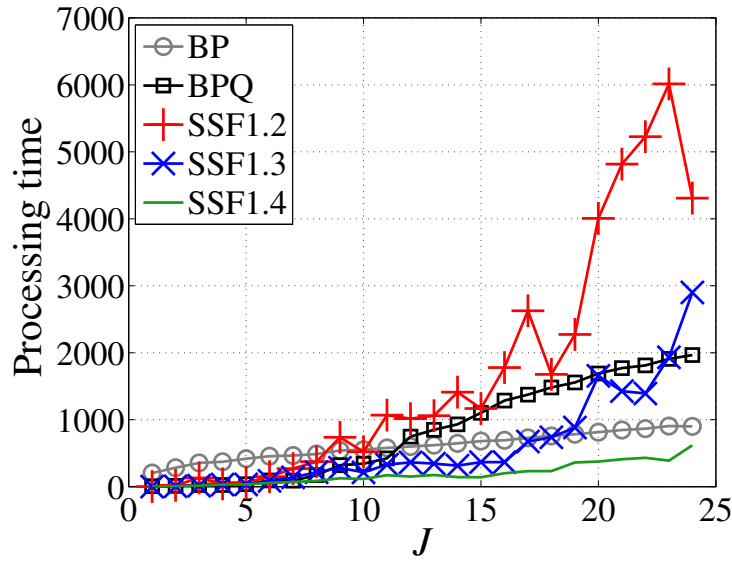


図 4.29: SSF1.4 の処理時間 (秒) (人工データ)

表 4.6: SSF1.4 の合計処理時間 (実データ)

	BP	BPQ	SSF1.2	SSF1.3	SSF1.4
合計処理時間 (hr:min:sec)	-	64:37:26	-	-	28:22:09

以外は全て1万回となった。しかし、SSF1.4は44%(966 / 2175)の探索を枝刈りすることにより、BPQよりも2倍以上速く処理が終了した。

4.3.4 考察

SSF1.4は特異領域上のHesse行列の固有値を基に探索の優先順位を決定し、探索数の上限を設定する手法により、解品質を劣化させることなく高速化できた。実データを用いた実験では、BPQの訓練誤差は $J = 18$ 以降も隠れユニットの増加に伴って単調減少しないことがあり、 $J = 18$ 以降の方が単調減少しないことが多かった。これは隠れユニットが増えるほど局所最適解や特異領域が増えることで、訓練誤差が比較的大きい局所最適解や特異領域に収束しやすくなり、学習が困難となるためと考えられる。一方SSF1.4の訓練誤差は単調減少し、人工データと実データを用いた実験においてBPQよりもテスト誤差が小さい解を得た。また、SSF1.4は負の固有値が小さい順に探索し、各 J での探索数が100回以下に制限することにより、人工データを用いた実験ではBPQよりも4倍以

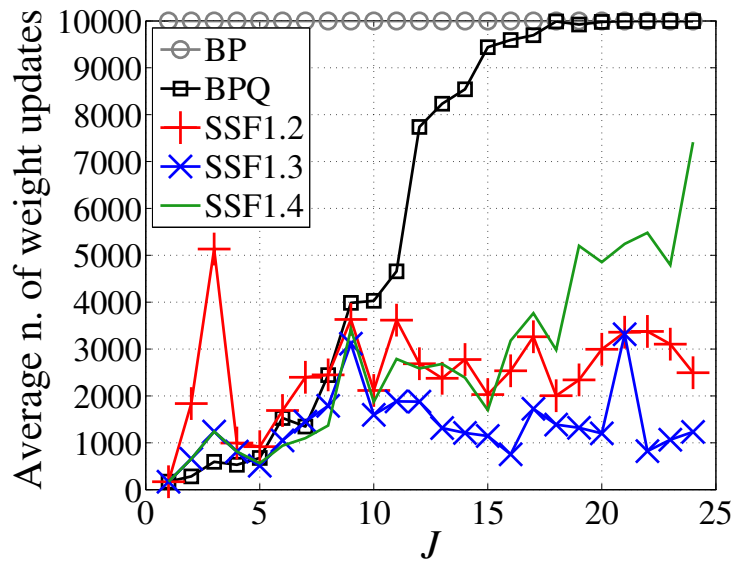


図 4.30: SSF1.4 の重み更新平均回数 (人工データ)

上, 実データでは2倍以上早く処理が終了した. ほとんどの場合 BPQ と SSF1.4 の探索数は同じであったにも関わらず, SSF1.4 の方が早く処理が終了したが, これは 67 % (2070 / 3106) の探索を枝刈りしたためと考えられる. また, 人工データを用いた実験では SSF1.4 の方が重み更新平均回数が少ない傾向もあった.

4.4 まとめ

本章ではまず 4.1 で, 学習が停滞する特異領域を逆に利用して実 MLP の解を探索する SSF1.2 を提案した. SSF1.2 は実 MLP($J-1$) の探索で得られた解に可約性写像を適用して形成した特異領域から実 MLP(J) の探索を行うことにより, 訓練誤差を単調に減少させ, 初期値をランダムに設定して BP や BPQ を用いて 100 回探索する方法よりも良質の解を得ることができた. しかし SSF1.2 では, 隠れユニットの増加に伴って特異領域が増えることで処理時間が大幅に増加した.

処理時間が大幅に増加する問題を解決するため, 4.2 では現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を導入した SSF1.3 を提案した. SSF1.3 は探索枝刈り手法により, 解品質を劣化させることなく高速化でき, 人工データを用いた実験では BPQ よりも速く処理が終了した. しかし, 実データを用いた実験では $J=5$ までは BP や BPQ よりも SSF1.3 の方が処理速度が速かったが, 隠れユニットの増加に伴って

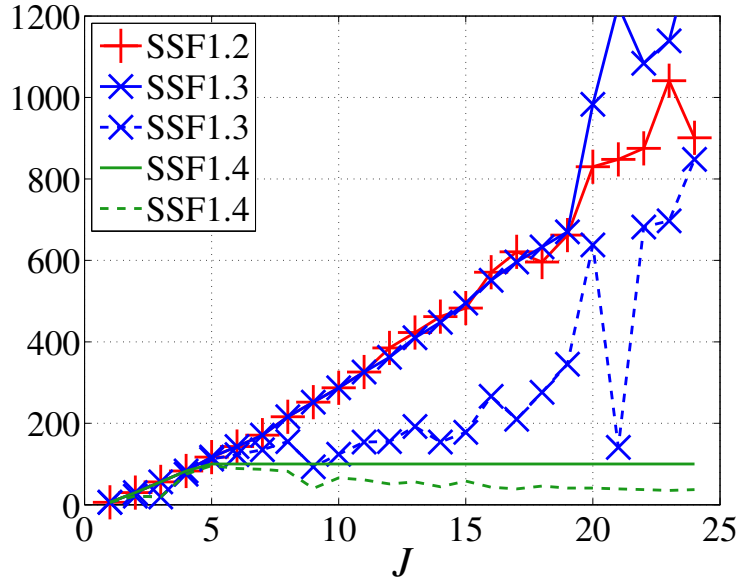


図 4.31: SSF1.4 の探索開始数（実線）と最後まで探索枝刈りされなかった探索数（点線）（人工データ）

特異領域が増え探索開始数が大幅に増加した．そのため多くの探索枝刈りを実行したとしても処理に時間がかかり BPQ よりも処理時間がかかった．

SSF1.3 をより高速化するため，4.3 では特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し，探索数の上限を設定する手法により一層高速化した SSF1.4 を提案した．SSF1.4 は各隠れユニット毎に実行する探索数に上限を設定し，特異領域上の Hesse 行列の固有値が小さい順に探索を行うことにより，解品質を劣化させることなく高速化できた．SSF1.4 の探索開始数は BPQ と同様 100 回としたが，探索枝刈り手法により人工データを用いた実験では BPQ の 4 倍以上，実データを用いた実験では 2 倍以上早く処理が終了した．また，BPQ では隠れユニットが多いほど訓練誤差が単調減少しない傾向が強くなったが，SSF1.4 では訓練誤差を単調減少させ，BPQ よりも小さいテスト誤差の解を得た．

今後の課題として，より多様なデータを用いて提案法を評価することが挙げられる．

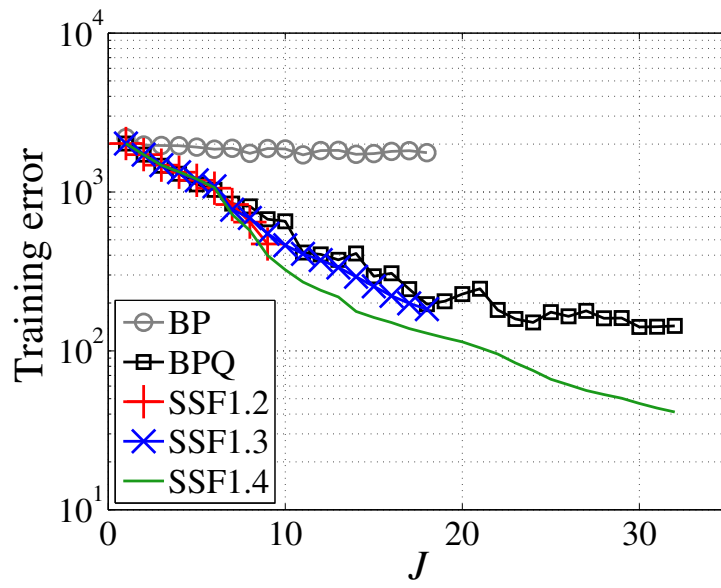


図 4.32: SSF1.4 の訓練誤差 (実データ)

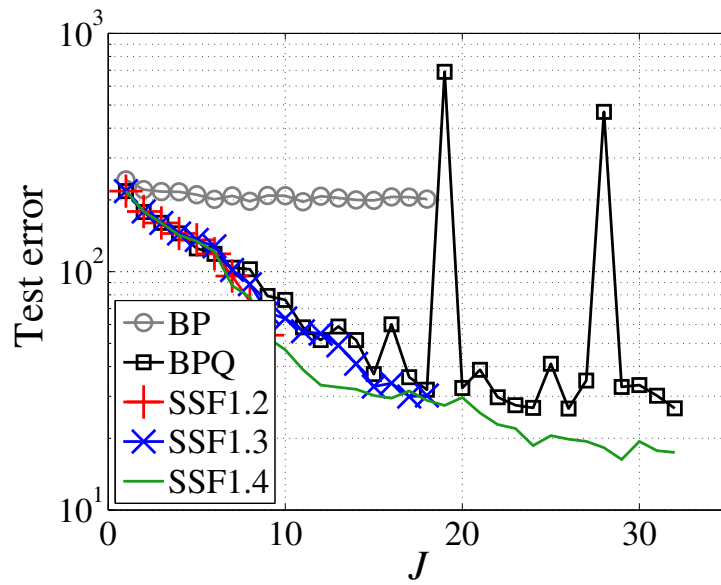


図 4.33: SSF1.4 のテスト誤差 (実データ)

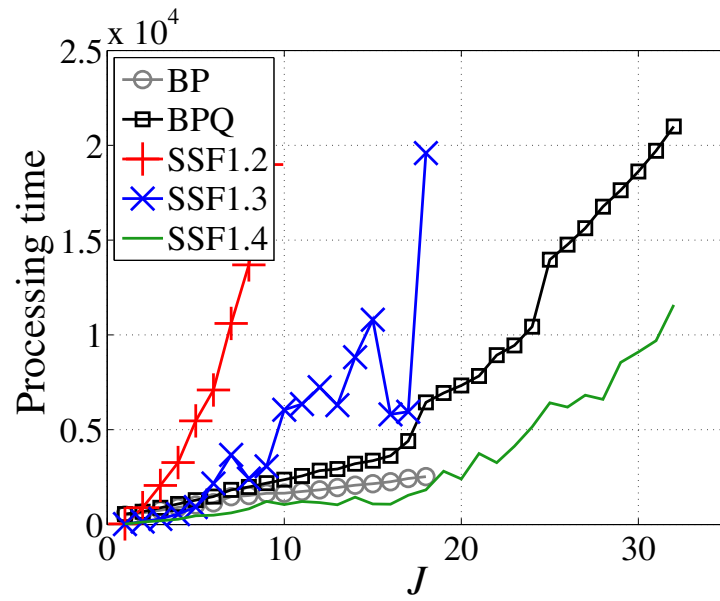


図 4.34: SSF1.4 の処理時間 (秒) (実データ)

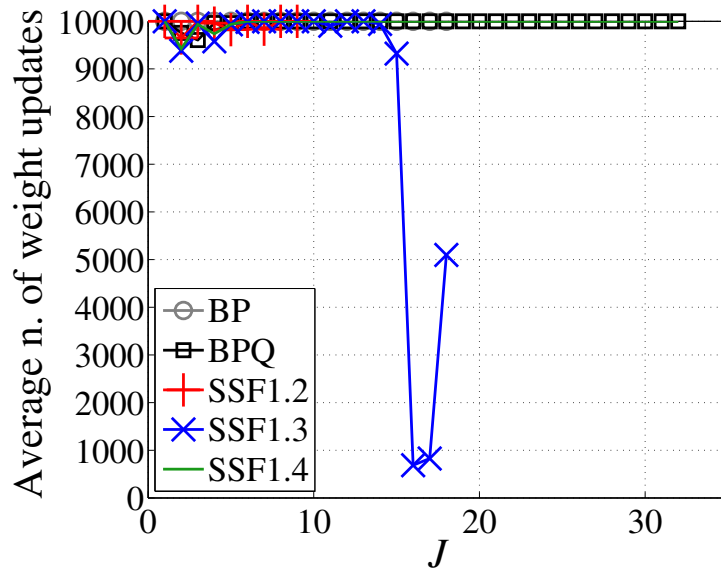


図 4.35: SSF1.4 の重み更新平均回数 (実データ)

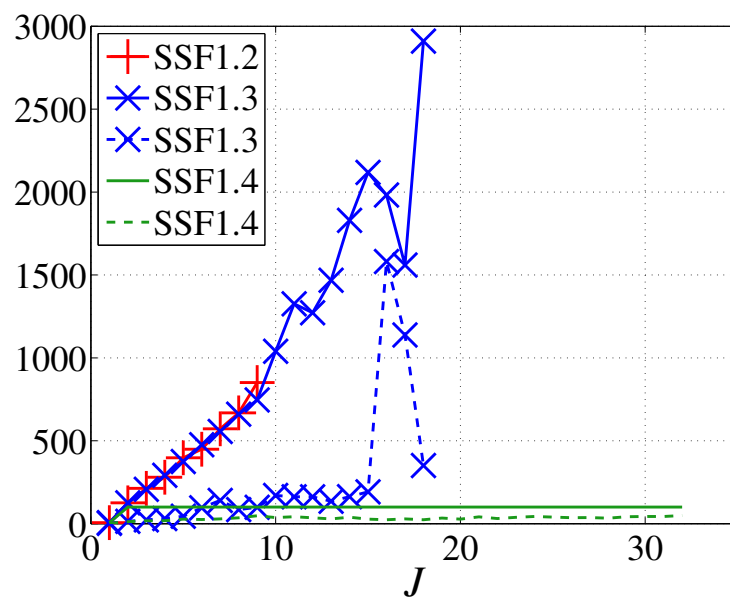


図 4.36: SSF1.4 の探索開始数（実線）と最後まで探索枝刈りされなかった探索数（点線）（実データ）

第5章 複素特異階段追跡法 (C-SSF)

3.5 で説明したように，複素 MLP の解に可約性写像を適用して形成される特異領域上のほとんどの点は，実 MLP と同様，降下するルートが存在する鞍点である．複素 MLP の学習においてもこの性質を利用し，学習が停滞する特異領域を利用して探索する複素特異階段追跡法 (C-SSF: Complex-valued Singularity Stairs Following) 1.0 を 5.1 で提案する．この方法は SSF1.2 を複素 MLP に拡張した方法であり，隠れユニットが $J - 1$ 個の複素 MLP (複素 MLP($J - 1$)) の最適解と入出力が等しい複素 MLP(J) の特異領域から探索空間を降下するため，隠れユニットの増加に伴って訓練誤差が単調減少する．

また，C-SSF1.0 においても，SSF1.2 と同様，探索空間の次元数は隠れユニットの増加に伴って増加し，その分特異領域も増えるため，隠れユニットの増加に伴って処理時間が増大する問題がある．そのため，現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を導入した SSF1.3 を複素 MLP 用に拡張した C-SSF1.1 を 5.2 で提案する．

更に，特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し，探索数の上限を設定する手法により一層高速化した SSF1.4 を複素 MLP 用に拡張した C-SSF1.3 を 5.3 で提案する．

5.1 C-SSF1.0

C-SSF1.0 では複素 MLP($J - 1$) の探索で得られた解に可約性写像を適用して形成した複素 MLP(J) の特異領域から探索を行う．以下ではまず特異領域から探索する方法につ

いて説明する.

5.1.1 特異領域からの探索

複素 MLP の特異領域から探索を開始することを考える. 実 MLP と同様, 特異領域は勾配がゼロであるため, 従来の勾配を利用した探索法では探索を開始できない. そのため, C-SSF1.0 においても最初の重みの更新のときのみ Hesse 行列の固有ベクトルを利用することを考える. ここで, Hesse 行列は $\mathbf{r}$ 表現とする. $\mathbf{r}$ 表現の Hesse 行列は実数対称行列であるため, 固有値と固有ベクトルも実数となる. そのため, SSF1.2 と同様, 負の固有値に対応する固有ベクトルの方向に重みを更新することで探索を開始できる. 固有ベクトル方向への探索幅も SSF1.2 と同様 2.2.5 で説明したバックトラッキング法を用いて決定した.

5.1.2 C-SSF1.0 の処理の概要

C-SSF1.0 の処理の概要を以下に示す. ただし, 考慮する隠れユニットの最大数を $J_{\max}$ とする.

C-SSF1.0 の処理の流れ

- 1: 隠れユニットが 0 個の複素 MLP の大域最適解を求める. ($w_0^{(0)} \rightarrow 1/N \sum_{\mu=1}^N y^\mu$)
 - 2: **for** $J = 1, \dots, J_{\max}$ **do**
 - 3: 複素 MLP($J - 1$) の探索で得られた最良解に可約性写像を適用して初期点を特異領域上に設定し, $\mathbf{r}$ 表現の Hesse 行列の固有値と固有ベクトルを求める.
 - 4: Step 3 で計算した固有値の中で負のものを選択し, その固有値に対応する固有ベクトル方向とその逆方向に探索空間を降下して解を探索する.
 - 5: **end for**
-

以下の計算機実験では Step 3 で利用する特異領域は $\Theta_J^{\alpha\beta}$ と Θ_J^γ とした. 初期点を $\Theta_J^{\alpha\beta}$ 上に設定する方法は以下とした. ただし, 複素 MLP(J) の $\mathbf{z}$ 表現の重みを $\{w_0^{(J)}, w_j^{(J)}, \mathbf{w}_j^{(J)}, j = 1, \dots, J\}$, 複素 MLP($J - 1$) の探索で得られた $\mathbf{z}$ 表現の重みを $\{w_0^{(J-1)}, w_j^{(J-1)}, \mathbf{w}_j^{(J-1)}, j =$

$1, \infty, J-1$ とする.

$$\begin{aligned} w_0^{(J)} &\rightarrow w_0^{(J-1)}, w_1^{(J)} \rightarrow 0, \mathbf{w}_1^{(J)} \rightarrow (p, 0, \infty, 0)^T, \\ w_j^{(J)} &\rightarrow w_{j-1}^{(J-1)}, \mathbf{w}_j \rightarrow \mathbf{w}_{j-1}^{(J-1)}, j=2, \infty, J \end{aligned}$$

p は複素数であるが, ここでは SSF1.2 と同様 $p = -1, 0, 1$ の 3 点とした. Θ_J^γ 上に設定する方法は以下とした. ただし, $m = 2, \infty, J$ とする.

$$\begin{aligned} w_0^{(J)} &\rightarrow w_0^{(J-1)}, w_1^{(J)} \rightarrow q \bullet w_{m-1}^{(J-1)}, w_m^{(J)} \rightarrow (1-q) \bullet w_{m-1}^{(J-1)}, \\ \mathbf{w}_1^{(J)} &\rightarrow \mathbf{w}_{m-1}^{(J-1)}, \mathbf{w}_m^{(J)} \rightarrow \mathbf{w}_{m-1}^{(J-1)}, w_j^{(J)} \rightarrow w_{j-1}^{(J-1)}, \mathbf{w}_j \rightarrow \mathbf{w}_{j-1}^{(J-1)}, j \in \{2, \infty, J\} \setminus \{m\} \end{aligned}$$

q も複素数であるが, ここでは $q = 0.5, 1.0, 1.5$ (それぞれ内分点, 内分と外分の境界点, 外分点) の $3 \bullet (J-1)$ 点とした. Step 4 の探索数は, SSF1.2 と同様, 特異領域上の降下するルートの数に依存する.

5.1.3 計算機実験

提案した C-SSF1.0 の性能を評価するため計算機実験を行った. 比較する従来法は, 3.4 で説明した直線探索付き C-BP (ここでは単に C-BP と呼ぶ) と準 Newton 法の一種である C-BFGS とした. C-BP と C-BFGS の初期重みは実部と虚部共に区間 $(-1, +1)$ の中からランダムに選択し, 各 J で 100 回の探索を行った. 各探索の終了条件は, 重み更新回数が 1 万回を越えるか, 探索幅が 10^{-16} 以下となった場合とした.

複素 MLP の活性化関数は 3.1.1 で説明した複素シグモイド関数を用い, 目的関数としては式 3.3 を用いた. プログラミング言語は MATLAB R2014a を用いた.

データとしては人工的に生成した 2 つのデータセットを用いて評価した. ここではそれぞれ人工データ 1, 人工データ 2 と呼ぶこととする.

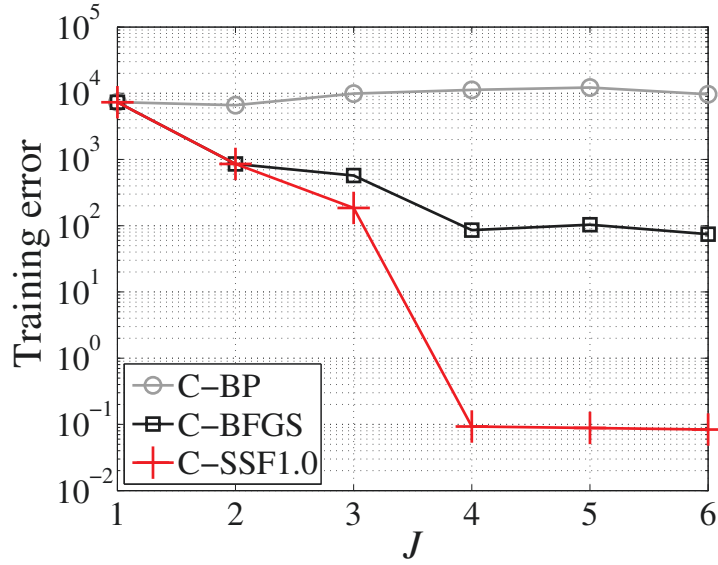


図 5.1: C-BP, C-BFGS, C-SSF1.0 の訓練誤差 (人工データ 1)

■ 人工データ 1 を用いた実験

人工データ 1 は以下の重みを持つ複素 MLP から生成した.

$$\begin{aligned}
 (w_0, w_1, w_2, w_3, w_4) &= (3 + 1i, 1 + 1i, 1 + 1i, 0 + 5i, 5 - 4i) \\
 (w_1, w_2, w_3, w_4) &= \begin{pmatrix} 2 + 3i, & 0 - 5i, & 4 - 5i, & 1 + 1i \\ 4 + 0i, & 2 + 2i, & 1 + 2i, & 2 + 2i \\ 3 + 1i, & 0 - 2i, & 4 - 4i, & 4 + 1i \\ 4 + 4i, & 1 + 4i, & 3 + 0i, & 4 - 1i \\ 0 - 5i, & 5 + 3i, & 1 - 5i, & 3 - 1i \\ 5 - 2i, & 4 + 2i, & 3 - 5i, & 5 + 4i \end{pmatrix}
 \end{aligned}$$

訓練データの入力信号は $x_1^\mu, \dots, x_5^\mu$ の 5 次元とした. $x_k^\mu (k = 1, \dots, 5)$ の実部と虚部は区間 $(0,1)$ の中でランダムに生成した. 教師信号 y^μ の値は上記の重みを持つ複素 MLP ($J = 4$) の出力の実部と虚部に標準偏差 0.01 の正規乱数を加え, 500 個のデータ点を生成した. テストデータは, x_k^μ を訓練データと同様の方法で訓練データとは別に生成し, y^μ には乱数を加えずに 1000 個のデータ点を生成した. 考慮する隠れユニットの最大数は $J_{\max} = 6$ とした.

図 5.1, 5.2 に各 J での最小の訓練誤差とそのときのテスト誤差を示す. C-BP では隠れユニット数を増やすと逆に訓練誤差が増加する傾向にあった. C-BFGS では $J = 5$ のと

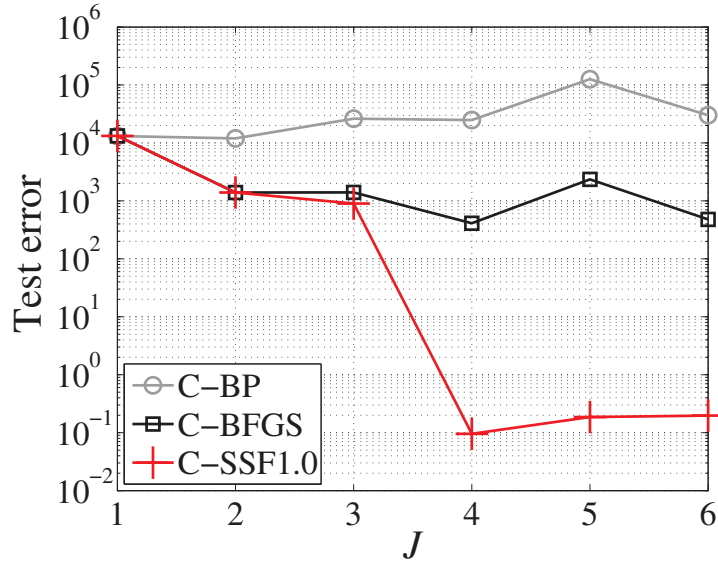


図 5.2: C-BP, C-BFGS, C-SSF1.0 のテスト誤差 (人工データ 1)

表 5.1: C-SSF1.0 の合計処理時間 (人工データ 1)

	C-BP	C-BFGS	C-SSF1.0
合計処理時間 (hr:min:sec)	07:23:24	00:11:19	00:34:27

き訓練誤差が増加した．C-SSF1.0 では，隠れユニットの増加に伴って訓練誤差が単調減少し， $J = 3$ 以降では C-BFGS よりも小さい訓練誤差の解を得た．また，テスト誤差は $J = 4$ のとき最小値を得た．人工データ 1 は複素 MLP($J = 4$) から生成したデータを用いたため，C-SSF1.0 は正解のモデルを選択できた．

図 5.3, 5.4 に各 J での処理時間と重み更新平均回数を，表 5.1 に合計処理時間を，図 5.5 に C-SSF1.0 の探索数を示す．C-BP は 7 時間以上かかったが，これは探索空間の条件数が大きいため，勾配のみを利用する C-BP では探索が困難となることが原因と考えられる．C-SSF1.0 の処理は $J = 4, 5, 6$ のとき C-BFGS よりも時間が掛かり，合計時間も C-BFGS よりも長かった．

■ 人工データ 2 を用いた実験

人工データ 2 は，以下の，対数螺旋の線形和の式から生成した．

$$y = \{0.001e^{0.1\phi} + 2.5e^{-0.1\phi} + 0.1e^{0.05\phi}\} e^{2i\phi} + e^{5i(\phi+\pi/3)} + e^{12i\phi} + e^{15i\phi}, \phi = 2\pi x(5.1)$$

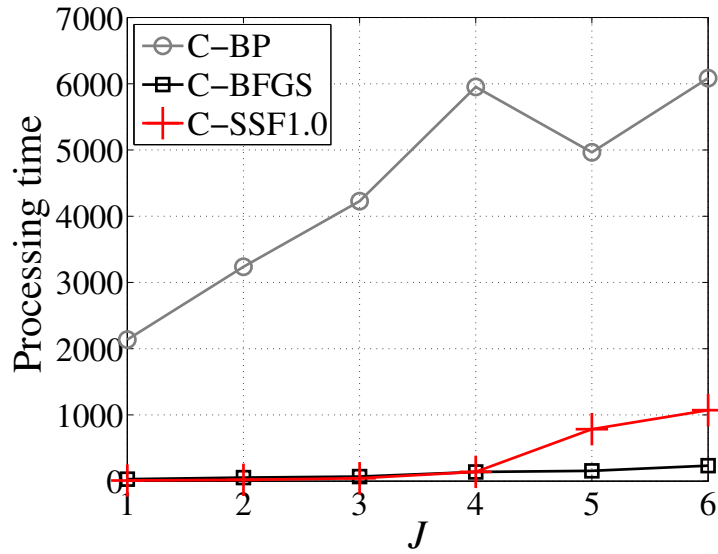


図 5.3: C-BP, C-BFGS, C-SSF1.0 の処理時間 (秒) (人工データ 1)

訓練データの入力信号 x^μ は実数区間 $(0, 10)$ の中でランダムに生成し, 教師信号 y^μ の実部と虚部に標準偏差が 0.01 の正規乱数を加え, 1000 個のデータ点を生成した. テストデータは x^μ の生成区間は訓練データとは異なり, 実数区間 $(10, 13)$ とし, y^μ に乱数を加えず 1000 個のデータ点を生成した. つまり, テストデータの x^μ の生成区間は未学習である. 考慮する隠れユニット数の最大は $J_{\max} = 16$ とした.

図 5.6, 5.7 に各 J での最小の訓練誤差とそのときのテスト誤差を示す. C-BP は J が増加しても訓練誤差が減少しておらず, テスト誤差も C-BFGS や C-SSF1.0 と比べて大きいことがわかる. C-BFGS は隠れユニット数が比較的大きい, $J = 9, 13, 14, 16$ のとき訓練誤差が増加した. これは実 MLP の探索法である BPQ の実験と同様, 隠れユニットが増えるほど局所最適解や特異領域が増えることで, 訓練誤差が比較的大きい局所最適解や特異領域に収束しやすくなったためと考えられる. C-SSF1.0 は, 隠れユニットの増加に伴って訓練誤差が単調減少し, $J = 5$ 以降は C-BFGS よりも訓練誤差が小さい解を得た. C-SSF1.0 では最小テスト誤差が C-BFGS よりも小さい解を得た.

C-BP, C-BFGS, C-SSF1.0 のテスト誤差が最小となったときの隠れユニット数は, それぞれ 15, 15, 12 であった. そのときの複素 MLP の出力を図 5.10 に示す. 図に示すように, C-BP はうまく近似できていないことがわかる. C-BFGS では, x が大きくなるにつれてずれが大きくなっている. 一方, C-SSF1.0 では $x = 13$ までうまく予測できていることがわかる.

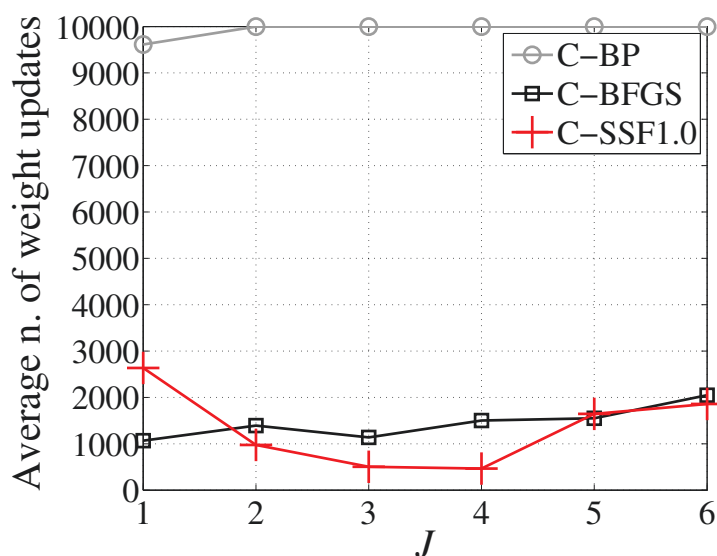


図 5.4: C-BP, C-BFGS, C-SSF1.0 の重み更新平均回数 (人工データ 1)

図 5.8, 5.9 に各 J での処理時間と重み更新平均回数を, 表 5.2 に合計処理時間を, 図 5.11 に C-SSF1.0 の探索数を示す. C-BP は 41 時間以上かかった. これは人工データを用いた実験と同様, 探索空間の条件数が大きいため C-BP では探索が困難となることが原因と考えられる. C-SSF1.0 の処理は $J = 10, \infty, 13$ のとき C-BFGS よりも大幅に時間が掛かった.

5.1.4 考察

5.1 では複素 $\text{MLP}(J-1)$ の最適解に可約性写像を適用して形成した特異領域から複素 $\text{MLP}(J)$ の探索を開始する C-SSF1.0 を提案した. 計算機実験では, 初期重みをランダムに設定し C-BP や C-BFGS を用いて解を探索する方法は隠れユニットを増やしても訓練誤差が減少しないことがあった. 一方, C-SSF1.0 は隠れユニットの増加に伴って訓練誤差を単調減少させ, 安定して良質の解を得ることができた. しかし, 隠れユニットの増加に伴って処理時間が増大する傾向があった.

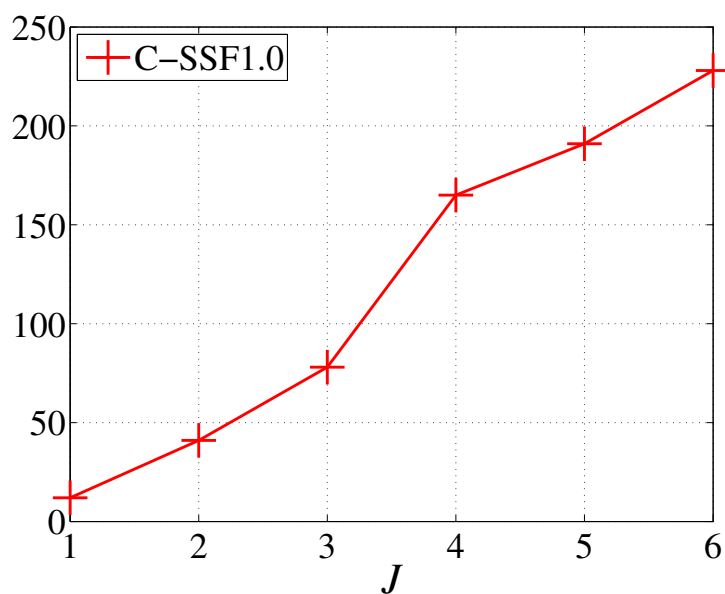


図 5.5: C-SSF1.0 の探索回数 (人工データ 1)

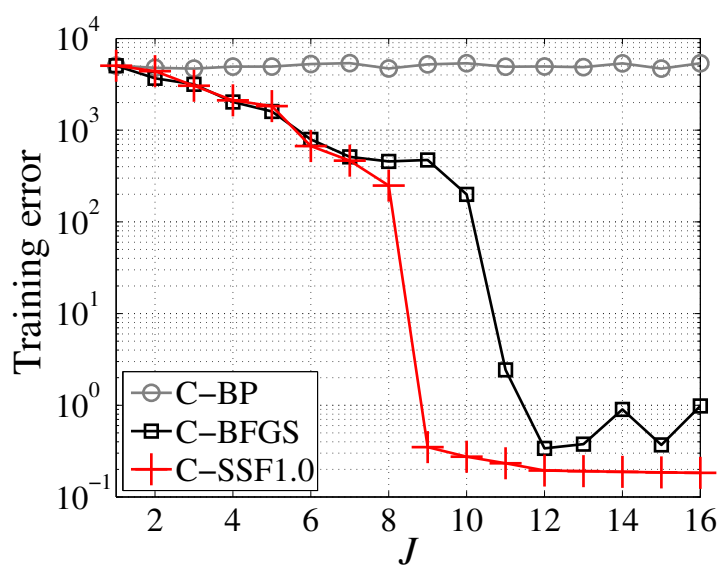


図 5.6: C-BP, C-BFGS, C-SSF1.0 の訓練誤差 (人工データ 2)

表 5.2: C-SSF1.0 の合計処理時間 (人工データ 2)

	C-BP	C-BFGS	C-SSF1.0
合計処理時間 (hr:min:sec)	41:09:15	03:34:55	14:46:59

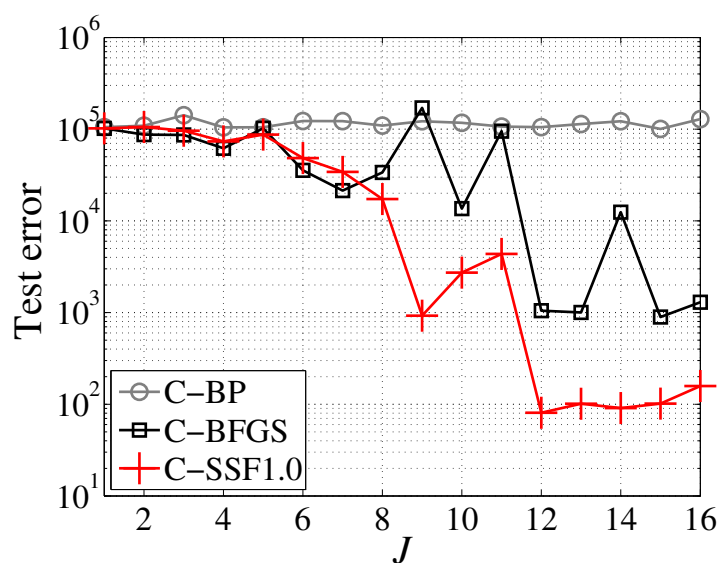


図 5.7: C-BP, C-BFGS, C-SSF1.0 のテスト誤差 (人工データ 2)

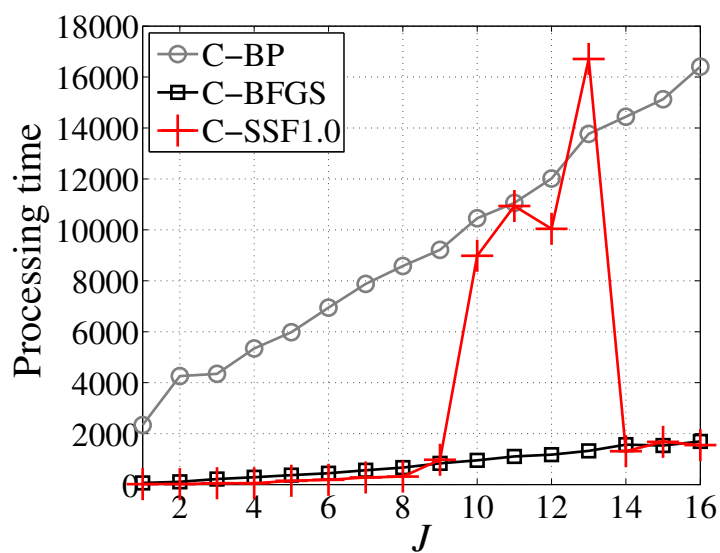


図 5.8: C-BP, C-BFGS, C-SSF1.0 の処理時間 (秒) (人工データ 2)

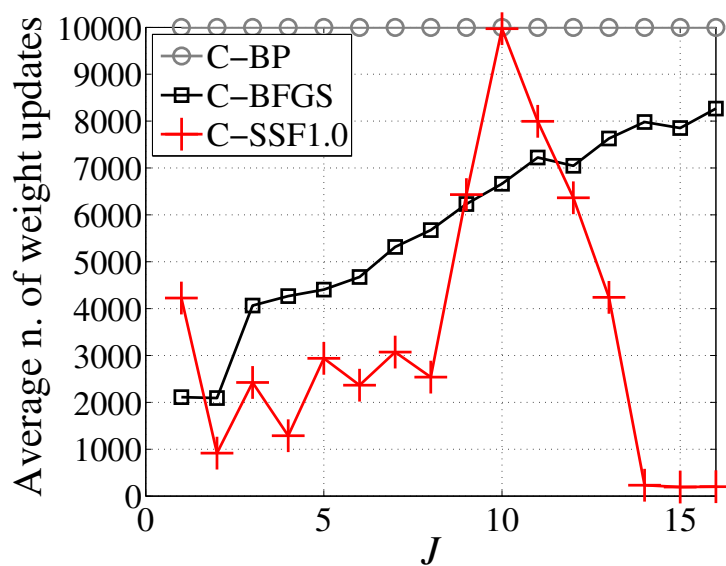


図 5.9: C-BP, C-BFGS, C-SSF1.0 の重み更新平均回数 (人工データ 2)

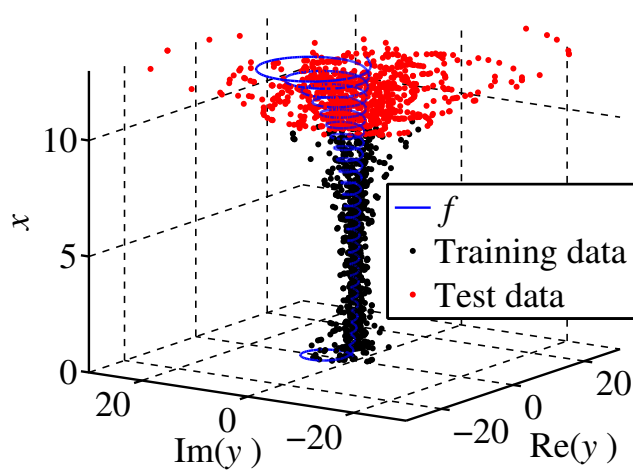
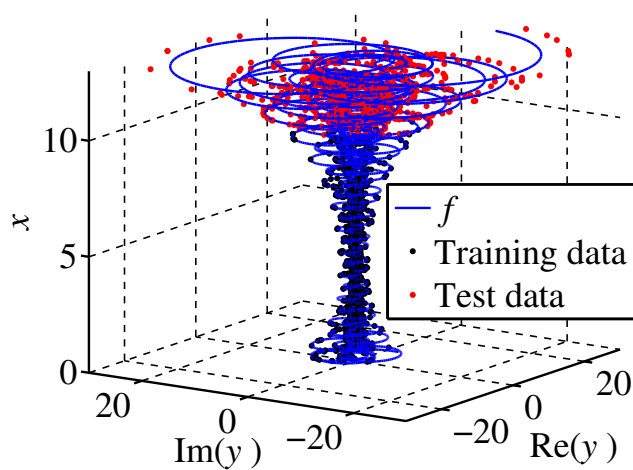
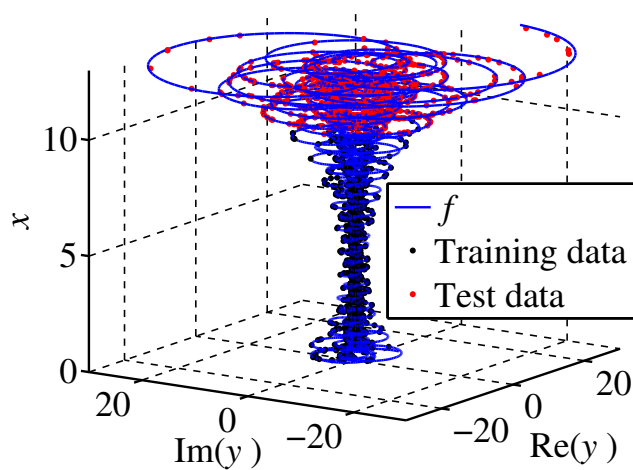
(a) C-BP($J = 15$)(b) C-BFGS($J = 15$)(c) C-SSF1.0($J = 12$)

図 5.10: C-BP, C-BFGS, C-SSF1.0 により得られた複素 MLP の出力 (人工データ 1)

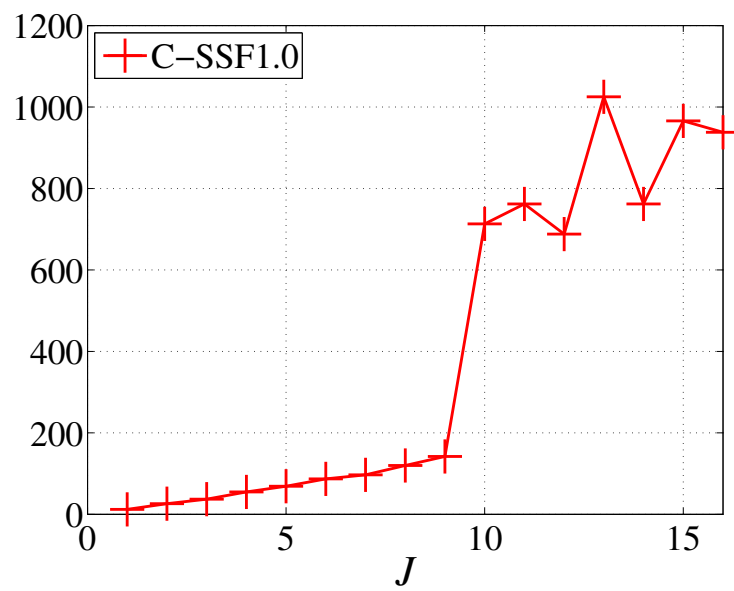


図 5.11: C-SSF1.0 の探索回数 (人工データ 2)

5.2 C-SSF1.1

C-SSF1.0は複素MLP($J-1$)の最適解に可約性写像を適用して形成された特異領域から探索空間を降下するため、訓練誤差を単調に減少させられる。しかし、SSF1.2と同様、隠れユニットの増加とともに探索空間の次元が増え、特異領域も増えるため、全ての特異領域から探索すると処理負荷が増大してしまう。そこで、現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法をC-SSF1.0に導入したC-SSF1.1を提案する。C-SSF1.1はSSF1.3を複素MLPに拡張した探索法である。

5.2.1 探索枝刈りによる高速化

探索枝刈り処理では、現在の探索が以前と同じ探索経路に合流したかどうかを判定し、合流したと判定した場合はその探索を枝刈りする。4.2の計算機実験で示したように、SSF1.3の処理時間は探索枝刈りによりを大幅に削減されたため、C-SSF1.0の探索においても有効に働くと考えられる。複素MLPの探索においては、 $\mathbf{r}$ 表現の重みを $\theta^{(t)}$ と $\phi^{(\tau)}$ と表現すれば4.2.1と同じ方法で探索枝刈りできる。ただし、実MLPと複素MLPの活性化関数は異なるため、以下で説明するように、複素MLPの活性化関数に適した前処理が必要となる。

■ 探索枝刈り判定の前処理

複素MLPの活性化関数として実験で用いる複素シグモイド関数を以下に再掲する。ただし、 $z = a + ib$, $i = \sqrt{-1}$ とする。

$$\sigma(z) = \frac{1}{1 + e^{-z}} = \frac{1 + e^{-a} \cos b + ie^{-a} \sin b}{1 + 2e^{-a} \cos b + e^{-2a}} \quad (5.2)$$

上記式の b は正弦関数と余弦関数にそのまま入力されるため、ある隠れユニット j への z 表現のバイアス w_{j0} を以下としても複素MLPの出力は変わらない。

$$w_{j0} \rightarrow \text{Re}(w_{j0}) + i\{\text{Im}(w_{j0}) \bmod 2\pi\} \quad (5.3)$$

また、式(5.2)の活性化関数は以下の式が成り立つ。

$$\sigma(z) = 1 - \sigma(-z) \quad (5.4)$$

そのため、ある隠れユニットから出力ユニットへ入力される信号は、以下の式が成り立つ。

$$w_j \sigma(\mathbf{w}_j^T \mathbf{x}) = w_j \quad w_j \sigma(\mathbf{w}_j^T \mathbf{x}) \quad (5.5)$$

この性質のため、以下のように重みを入れ替えても複素 MLP の出力は変わらない。

$$w_0 \rightarrow w_0 \quad w_j, \quad w_j \rightarrow w_j, \quad \mathbf{w}_j \rightarrow \mathbf{w}_j \quad (5.6)$$

そのため、探索枝刈り判定を行う前に、式 (5.3) を行い、 $\text{Re}(w_j)$ の符号を正にそろえ、上記の活性化関数の性質を考慮した探索の枝刈りを行う。

また、隠れユニットの並びは、活性化関数の種類に依らず、順序を入れ替えても複素 MLP の出力は変わらない。そのため、式 (5.3) を行い、 $\text{Re}(w_j)$ の符号をそろえた後、 $\text{Re}(w_j)$ が昇順となるように隠れユニットの順番を入れ替える前処理を行う。この処理により、隠れユニットの並びを考慮した探索の枝刈りが可能となる。

ただし、これらの重みの変換は探索枝刈り判定のためにのみ行う。そのため、変換後の重みを次の探索の判定のために記録するが、実際の探索では変換後の重みを反映させない。

5.2.2 計算機実験

提案した C-SSF1.3 の性能を 5.1.3 と同様のデータを用いて評価した。

■ 人工データ 1 を用いた実験

5.1.3 と同じ人工データ 1 を用いて実験を行った。図 5.12, 5.13 に各 J での最小の訓練誤差とそのときのテスト誤差を示す。C-SSF1.1 では、C-SSF1.0 と同様、訓練誤差が単調減少し、 $J = 4$ のとき最小のテスト誤差の解を得た。

図 5.14, 5.15 に各 J での処理時間と重み更新平均回数を、表 5.3 に合計処理時間を、図 5.16 に SSF1.3 の探索開始数と最後まで探索枝刈りされなかった探索数を示す。ただし、SSF1.3 の重み更新平均回数は探索枝刈りされなかった探索の重み更新回数の平均とした。C-SSF1.1 の探索開始数は C-SSF1.0 とほとんど同じであったが、C-SSF1.1 では 49 % (325 / 661) の探索を枝刈りすることにより、処理時間を削減し、C-BFGS とほぼ同じ処理時間となった。

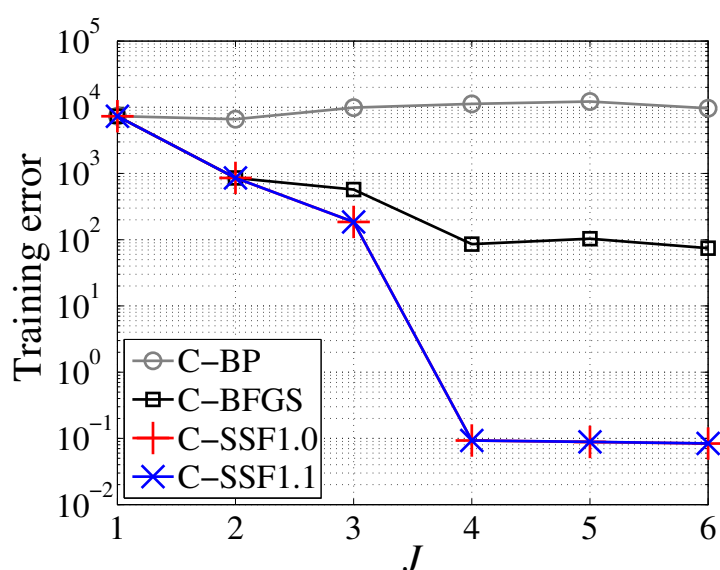


図 5.12: C-SSF1.1 の訓練誤差 (人工データ 1)

表 5.3: C-SSF1.1 の合計処理時間 (人工データ 1)

	C-BP	C-BFGS	C-SSF1.0	C-SSF1.1
合計処理時間 (hr:min:sec)	07:23:24	00:11:19	00:34:27	00:11:06

■ 人工データ 2 を用いた実験

5.1.3 と同じ人工データ 2 を用いて実験を行った。図 5.17, 5.18 に各 J での最小の訓練誤差とそのときのテスト誤差を示す。C-SSF1.1 は、C-SSF1.0 と同様、訓練誤差を単調減少させ、C-BFGS よりも小さいテスト誤差の解を得た。

C-SSF1.3 のテスト誤差が最小となったときの隠れユニット数は、16 であった。そのときの複素 MLP の出力を図 5.21 に示す。C-SSF1.1 では、C-SSF1.0 と同様、 $x = 13$ までうまく予測できていることがわかる。

図 5.19, 5.20 に各 J での処理時間と重み更新平均回数を、表 5.4 に合計処理時間を、図 5.22 に C-SSF1.1 の探索開始数と最後まで探索枝刈りされなかった探索数を示す。ただし、C-SSF1.1 の重み更新平均回数は探索枝刈りされなかった探索の重み更新回数の平均とした。C-SSF1.1 の重み更新平均回数は $J = 1, 10$ 以外は全て C-BFGS よりも少なくなった。また、C-SSF1.1 では 59 % (3752 / 6348) の探索を枝刈りすることにより処理時間を大幅に削減でき、処理速度は C-SSF1.0 の 4 倍となった。そのため処理時間は C-BFGS とほとんど同じ程度となった。

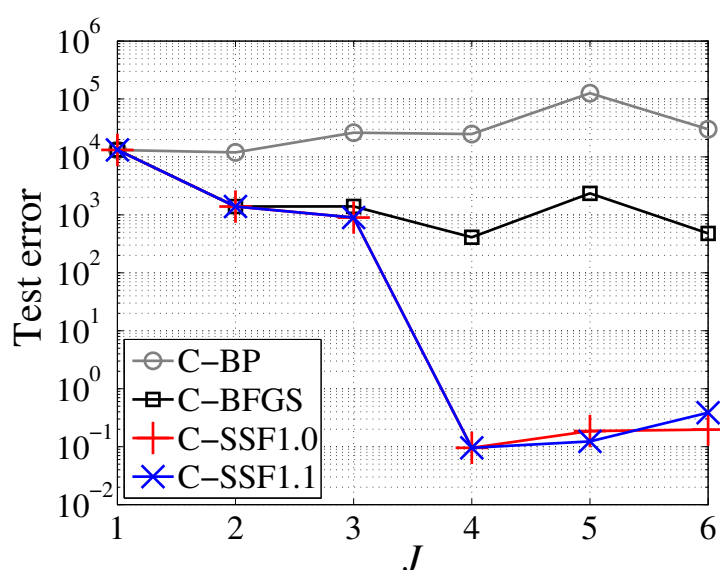


図 5.13: C-SSF1.1 のテスト誤差 (人工データ 1)

表 5.4: C-SSF1.1 の合計処理時間 (人工データ 2)

	C-BP	C-BFGS	C-SSF1.0	C-SSF1.1
合計処理時間 (hr:min:sec)	41:09:15	03:34:55	14:46:59	03:41:26

5.2.3 考察

C-SSF1.1 は、現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を C-SSF1.0 に導入することにより、解品質を劣化させることなく高速化できた。人工データ 1 を用いた実験では C-SSF1.0 と同じように最小のテスト誤差を $J = 4$ のときに得るとともに、処理時間も大幅に高速化され、C-BFGS とほぼ同じとなった。また、人工データ 2 を用いた実験においても C-BFGS よりも良質の解を得るとともに、処理速度は C-SSF1.0 の 4 倍に高速化され、C-BFGS とほぼ同じ処理時間となった。

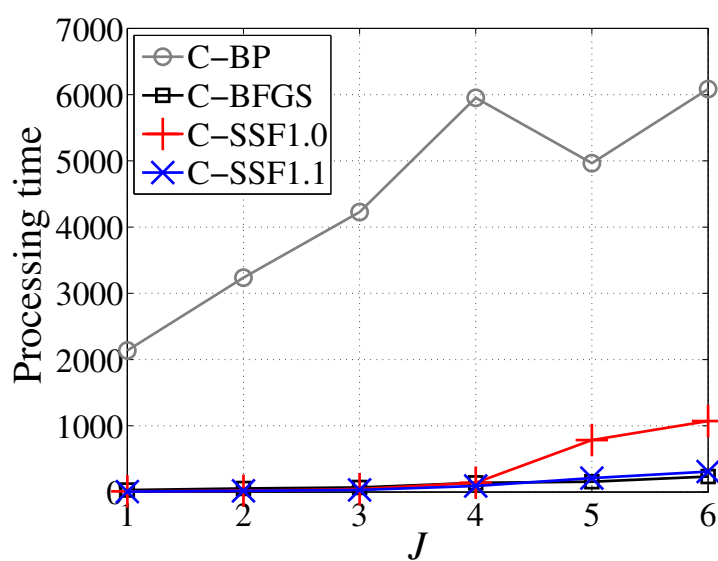


図 5.14: C-SSF1.1 の処理時間 (秒) (人工データ 1)

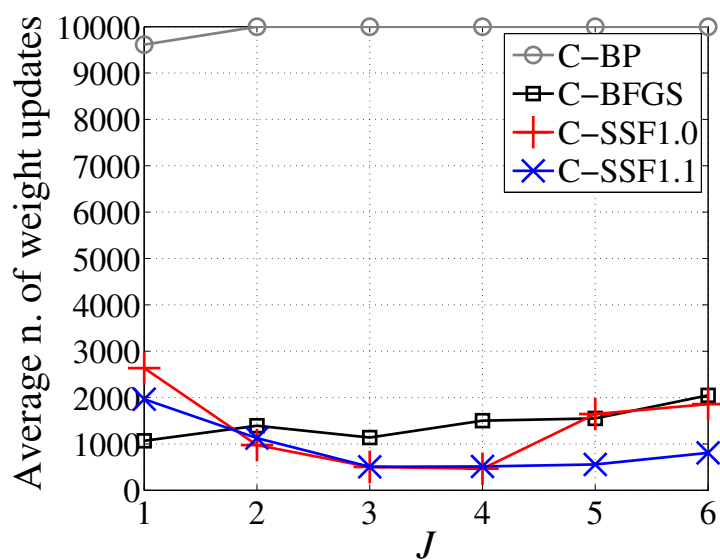


図 5.15: C-SSF1.1 の重み更新平均回数 (人工データ 1)

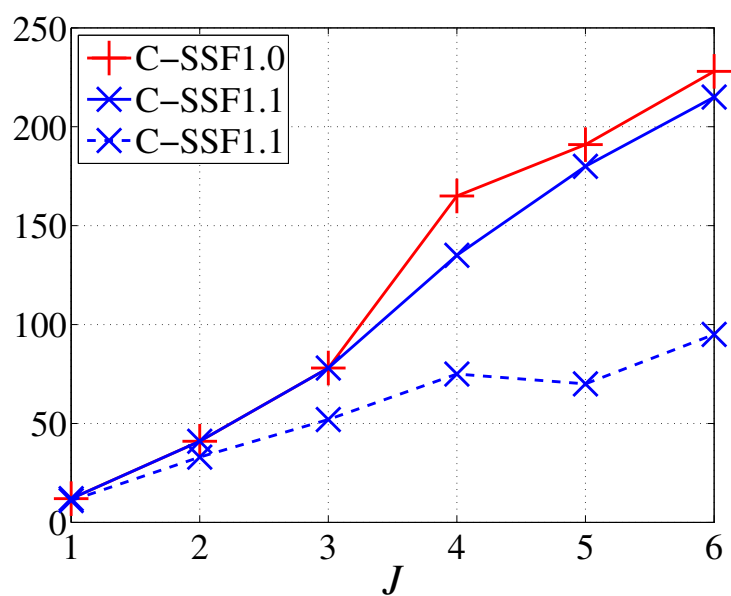


図 5.16: C-SSF1.1 の探索開始数（実線）と探索枝刈りされなかった探索回数（点線）（人工データ 1）

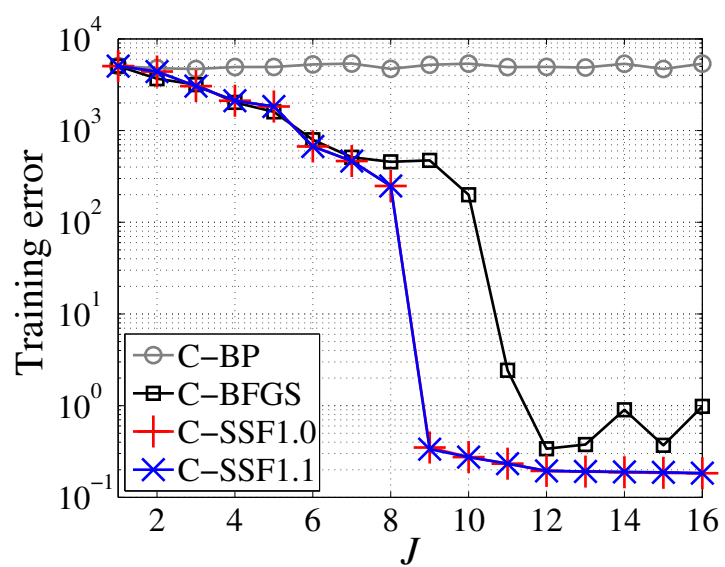


図 5.17: C-SSF1.1 の訓練誤差（人工データ 2）

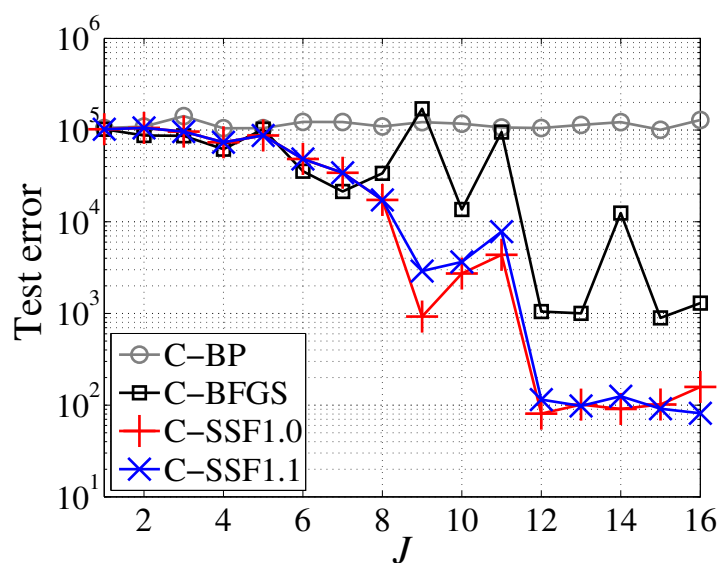


図 5.18: C-SSF1.1 のテスト誤差 (人工データ 2)

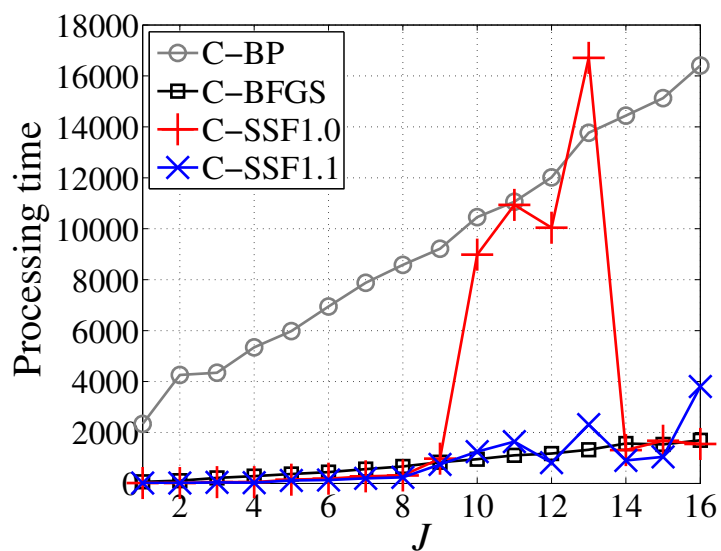


図 5.19: C-SSF1.1 の処理時間 (秒) (人工データ 2)

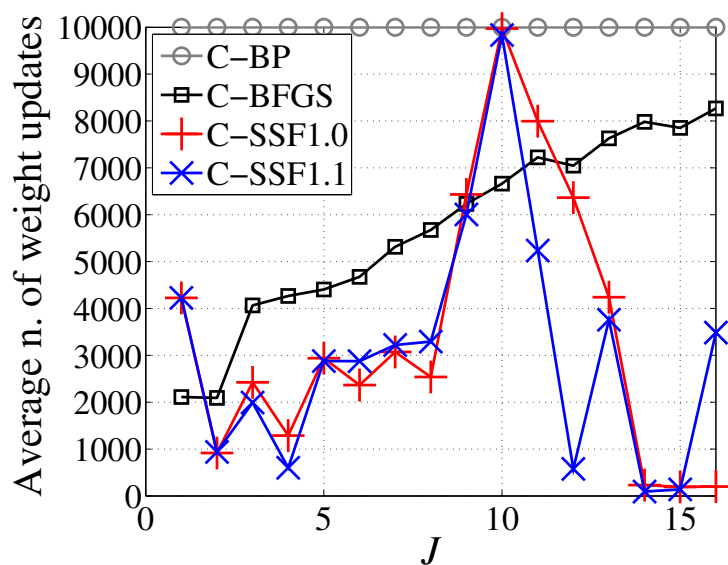


図 5.20: C-SSF1.1 の重み更新平均回数 (人工データ 2)

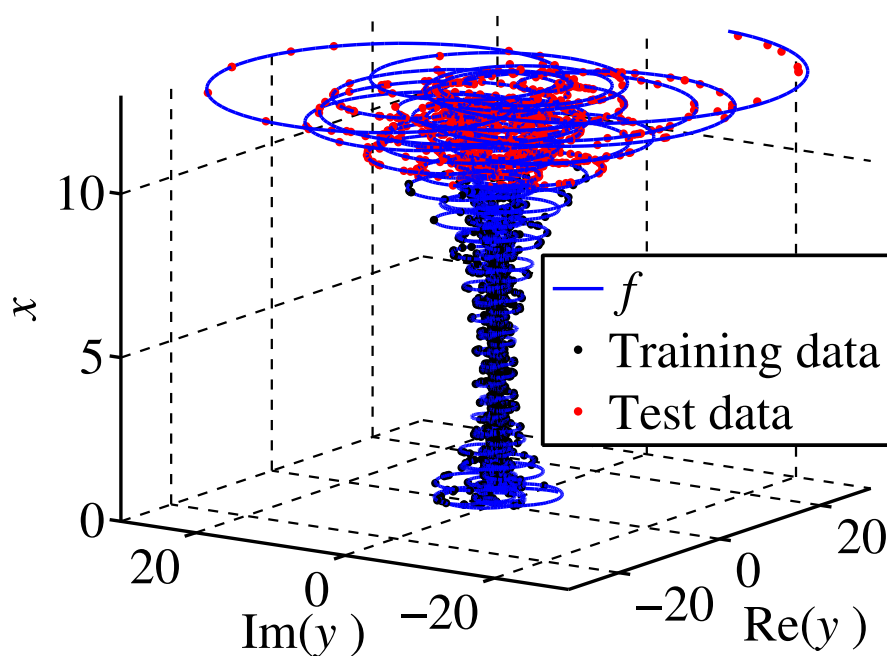


図 5.21: C-SSF1.1 により得られた複素 MLP($J = 16$) の出力 (人工データ 2)

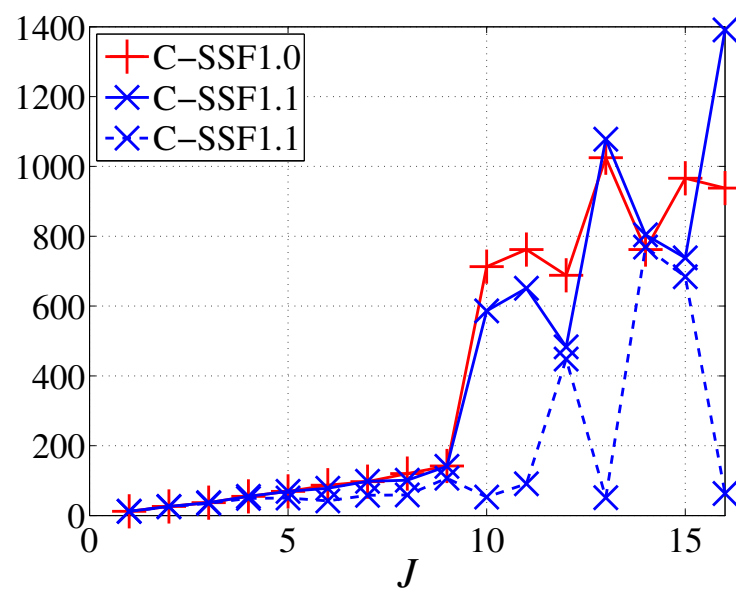


図 5.22: C-SSF1.1 の探索開始数（実線）と探索枝刈りされなかった探索回数（点線）（人工データ 2)

5.3 C-SSF1.3

C-SSF1.1 は C-SSF1.0 に探索枝刈りを導入することにより高速化され、C-BFGS と同程度の処理時間となった。5.3 では C-SSF1.1 を更に高速化した C-SSF1.3 を提案する。C-SSF1.3 は、各隠れユニット毎に実行する探索数に上限を設定し、特異領域上の Hesse 行列の固有値が小さい順に探索を行うことで高速化した SSF1.4 を複素 MLP に拡張した探索法である。この高速化手法により、解品質を劣化させることなく C-BFGS よりも高速に処理することを目指す。

探索本数削減による高速化は、C-SSF1.3 では $\mathbf{r}$ 表現の Hesse 行列を用いるため、4.3.1 で説明した手法をそのまま用いることができる。

5.3.1 C-SSF1.3 の処理の概要

C-SSF1.3 の処理の概要を以下に示す。ただし、考慮する隠れユニットの最大数を $J_{\max}$ とする。

C-SSF1.3 の処理の流れ

- 1: 隠れユニットが 0 個の複素 MLP の大域最適解を求める。($w_0^{(0)} \rightarrow 1/N \sum_{\mu=1}^N y^\mu$)
 - 2: **for** $J = 1, \dots, J_{\max}$ **do**
 - 3: 複素 MLP($J - 1$) の探索で得られた最良解に可約性写像を適用して初期点を特異領域上に設定し、 $\mathbf{r}$ 表現の Hesse 行列の固有値と固有ベクトルを求める。
 - 4: **while** 終了条件を満たすまで以下を繰り返す。 **do**
 - 5: Step 3 で計算した負の固有値の中で最も小さい固有値から順に選択し、その固有値に対応する固有ベクトル方向とその逆方向に探索空間を降下して解を探索する。
 - 6: **end while**
 - 7: **end for**
-

以下の計算機実験では Step 3 で利用する特異領域は C-SSF1.0 と同じとした。Step 5 の終了条件は探索枝刈りされた探索も含めて探索数が 100 となるか、選択する負の固有値が無くなったときとした。

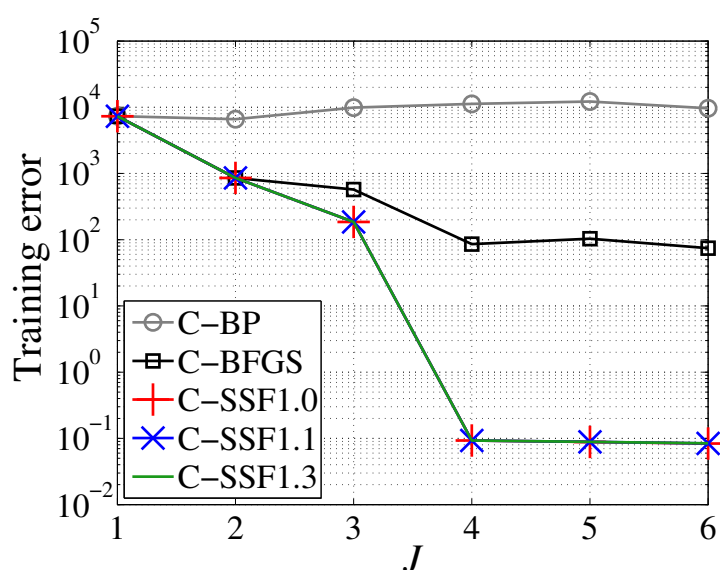


図 5.23: C-SSF1.3 の訓練誤差 (人工データ 1)

表 5.5: C-SSF1.3 の合計処理時間 (人工データ 1)

	C-BP	C-BFGS	C-SSF1.0	C-SSF1.1	C-SSF1.3
合計処理時間 (hr:min:sec)	07:23:24	00:11:19	00:34:27	00:11:06	00:08:04

5.3.2 計算機実験

提案した C-SSF1.3 の性能を 4.1.3 と同様のデータを用いて評価した.

■ 人工データ 1 を用いた実験

5.1.3 と同じ人工データ 1 を用いて実験を行った. 図 5.23, 5.24 に各 J での最小の訓練誤差とそのときのテスト誤差を示す. C-SSF1.3 では, C-SSF1.0, C-SSF1.1 と同様, 訓練誤差が単調減少し, $J = 4$ のとき最小のテスト誤差の解を得た.

図 5.25, 5.26 に各 J での処理時間と重み更新平均回数を, 表 5.5 に合計処理時間を, 図 5.27 に C-SSF1.3 の探索開始数と最後まで探索枝刈りされなかった探索数を示す. ただし, C-SSF1.3 の重み更新平均回数は, C-SSF1.1 と同様, 探索枝刈りされなかった探索の重み更新回数の平均とした. C-SSF1.3 の探索開始数は, $J = 4$ 以降全て上限の 100 に達したが, 解品質は C-SSF1.0, 1.1 と同様良質の解を得た. また, C-SSF1.3 は探索開始数を 100 回に制限することで合計処理時間が最短となった. $J = 4$ 以降の C-SSF1.3 の探索開始数

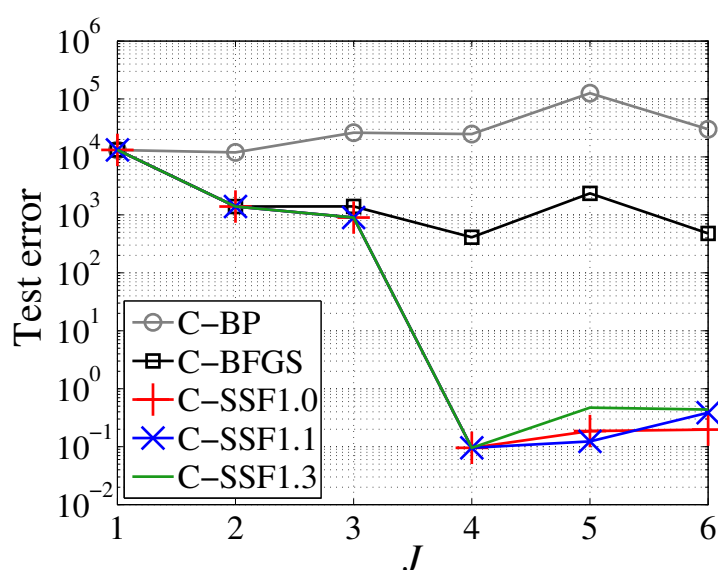


図 5.24: C-SSF1.3 のテスト誤差 (人工データ 1)

は, C-BP や C-BFGS と同様, 100 であったにも関わらず, 隠れユニット数が 1 から 6 まですべて C-SSF1.3 の方が C-BFGS よりも速かった. これは C-SSF1.3 では 58 % (252 / 431) の探索を枝刈りすることにより高速化されたことに加え, $J = 2$ 以降の C-SSF1.3 の重み更新平均回数が C-BFGS よりも少ないことが原因と考えられる.

■ 人工データ 2 を用いた実験

5.1.3 と同じ人工データ 2 を用いて実験を行った. 図 5.28, 5.29 に各 J での最小の訓練誤差とそのときのテスト誤差を示す. C-SSF1.3 は, C-SSF1.0, 1.1 と同様, 訓練誤差を単調減少させ, C-BFGS よりも小さいテスト誤差の解を得た.

C-SSF1.3 のテスト誤差が最小となったときの隠れユニット数は, 12 であった. そのときの複素 MLP の出力を図 5.32 に示す. C-SSF1.3 では, C-SSF1.0, 1.1 と同様, $x = 13$ まですぐうまく予測できていることがわかる.

図 5.30, 5.31 に各 J での処理時間と重み更新平均回数を, 表 5.6 に合計処理時間を, 図 5.33 に C-SSF1.3 の探索開始数と最後まで探索枝刈りされなかった探索数を示す. ただし, C-SSF1.3 の重み更新平均回数は, C-SSF1.1 と同様, 探索枝刈りされなかった探索の重み更新回数の平均とした. C-SSF1.3 の重み更新平均回数は $J = 1, 9, 10, 11$ 以外は全て C-BFGS よりも少なくなり, かつ C-SSF1.3 では 44 % (555 / 1275) の探索を枝刈りした.

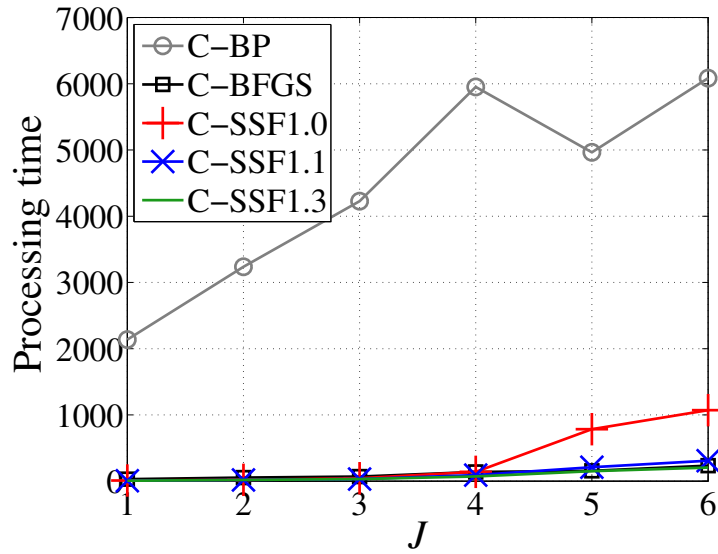


図 5.25: C-SSF1.3 の処理時間 (秒) (人工データ 1)

表 5.6: C-SSF1.3 の合計処理時間 (人工データ 2)

	C-BP	C-BFGS	C-SSF1.0	C-SSF1.1	C-SSF1.3
合計処理時間 (hr:min:sec)	41:09:15	03:34:55	14:46:59	03:41:26	01:27:58

その結果 C-SSF1.3 の処理速度は最も速く、C-BFGS より 2 倍以上速く処理が終了した。

5.3.3 考察

C-SSF1.3 は探索数上限設定により大幅に高速化され、最も速く処理が終了した。また、探索の優先順位を固有値が小さい順とすることで、解品質を劣化させることなく高速化でき、人工データ 1 を用いた実験と人工データ 2 を用いた実験どちらも処理時間は最短となった。人工データ 2 を用いた実験では C-BFGS より 2 倍以上速く処理が終了した。

5.4 まとめ

本章ではまず 5.1 で、学習が停滞する特異領域を逆に利用して複素 MLP の解を探索する C-SSF1.0 を提案した。C-SSF1.0 は複素 MLP($J-1$) の探索で得られた解に可約性写像を適用して形成した特異領域から複素 MLP(J) の探索を行うことにより、訓練誤差を単

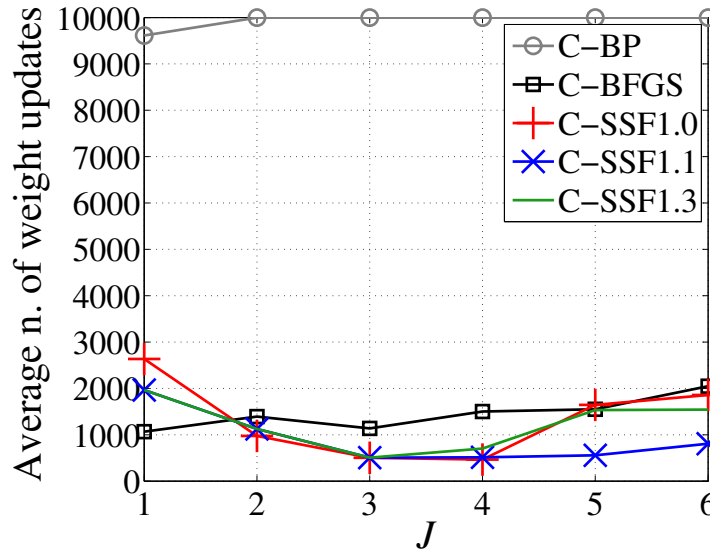


図 5.26: C-SSF1.3 の重み更新平均回数（人工データ 1）

調に減少させ、初期値をランダムに設定して C-BP や C-BFGS を用いて 100 回探索する方法よりも良質の解を得ることができた。しかし C-SSF1.0 では、SSF1.2 と同様、隠れユニットの増加に伴って特異領域が増えることで処理時間が大幅に増加した。

処理時間が大幅に増加する問題を解決するため、5.2 では現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を導入した C-SSF1.1 を提案した。C-SSF1.1 は探索枝刈り手法により、解品質を劣化させることなく高速化でき、人工データ 1 を用いた実験と人工データ 2 を用いた実験では C-BFGS とほとんど同じ処理時間となった。

C-SSF1.1 をより高速化するため、5.3 では特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し、探索数の上限を設定する手法により一層高速化した C-SSF1.3 を提案した。C-SSF1.3 は各隠れユニット毎に実行する探索数に上限を設定し、特異領域上の Hesse 行列の固有値が小さい順に探索を行うことにより、解品質を劣化させることなく高速化できた。C-SSF1.3 の探索開始数は C-BFGS と同様 100 回としたが、C-SSF1.3 の方が早く収束する傾向があり、かつ探索枝刈り手法により人工データ 1 を用いた実験では C-BFGS よりも速く処理が終了し、人工データ 2 を用いた実験では C-BFGS の 2 倍以上早く処理が終了した。また、第 4 章の計算機実験で用いた、重みの初期値をランダムに設定して BPQ を用いて 100 回探索する方法と同様、C-BFGS では隠れユニットが多いほど訓練誤差が単調減少しない傾向が強くなったが、C-SSF1.3 では訓練誤差を単調減少させ、C-BFGS よりも小さいテスト誤差の解を得た。

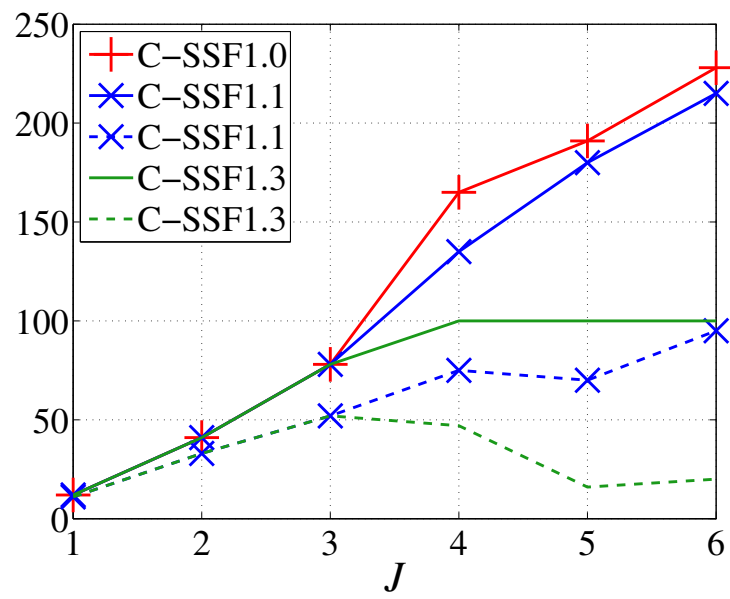


図 5.27: C-SSF1.3 の探索開始数（実線）と探索枝刈りされなかった探索回数（点線）（人工データ 1)

今後の課題として、より多様なデータを用いて提案法を評価することが挙げられる。

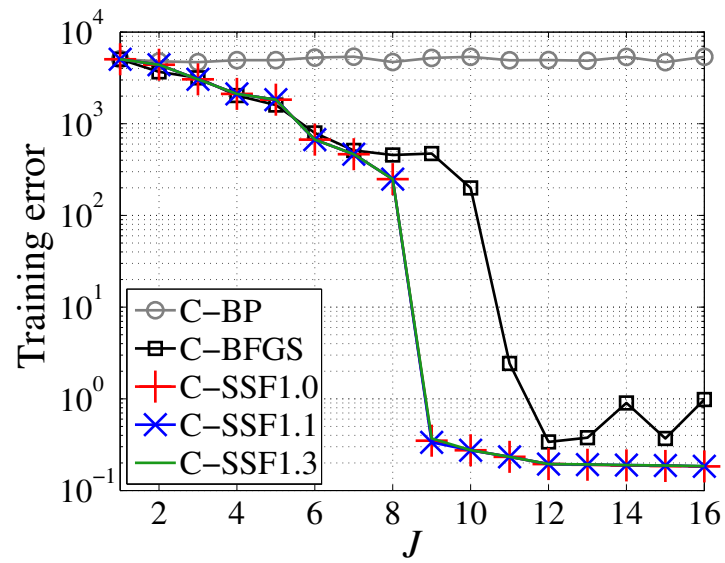


図 5.28: C-SSF1.3 の訓練誤差 (人工データ 2)

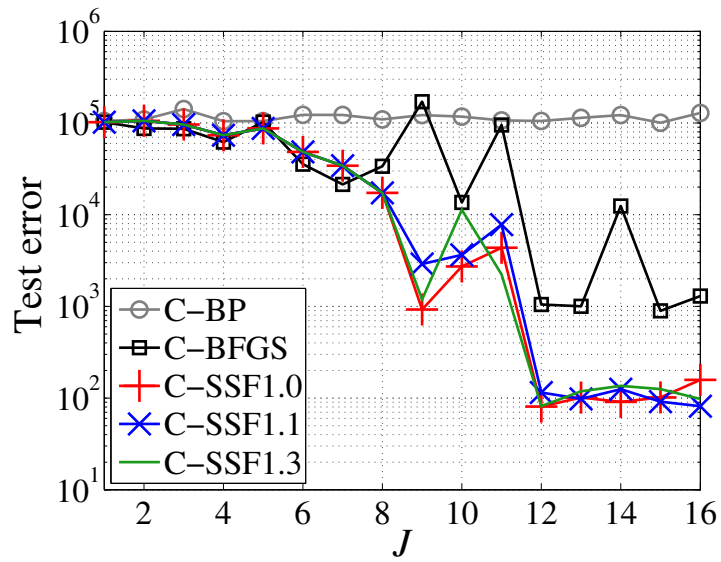


図 5.29: C-SSF1.3 のテスト誤差 (人工データ 2)

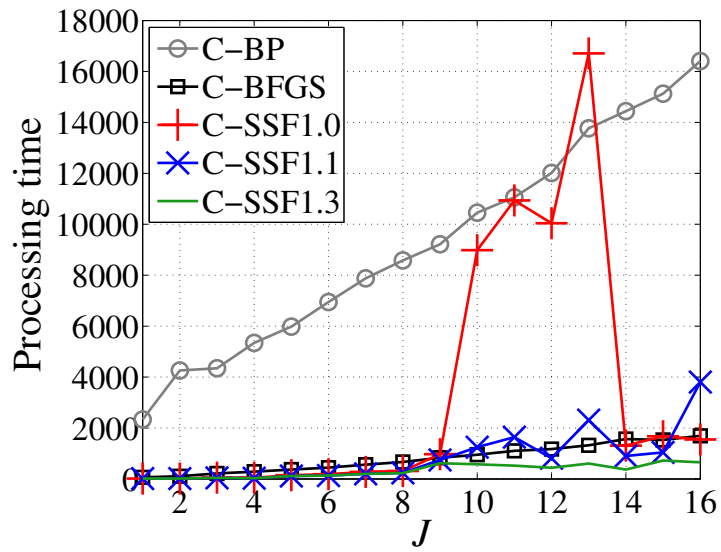


図 5.30: C-SSF1.3 の処理時間 (秒) (人工データ 2)

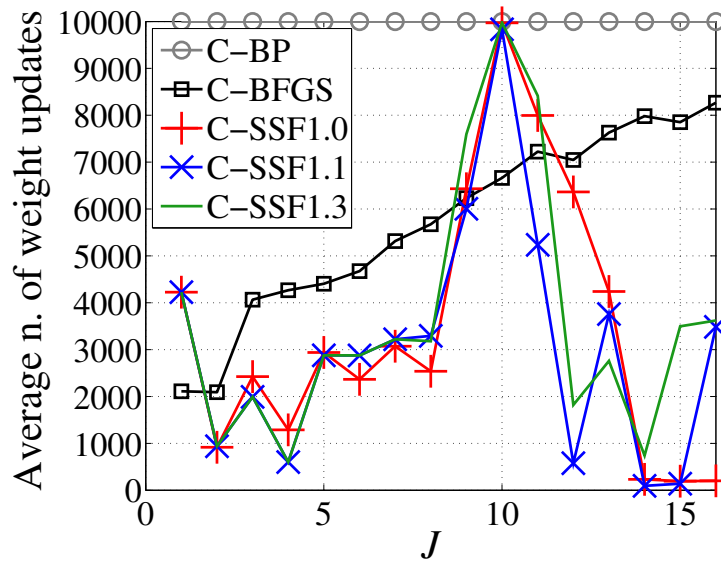


図 5.31: C-SSF1.3 の重み更新平均回数 (人工データ 2)

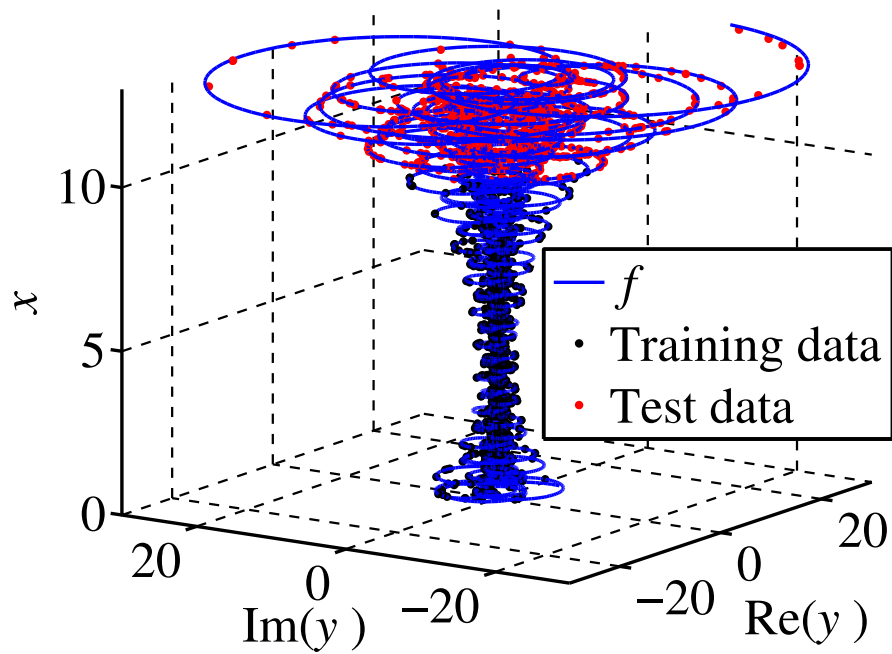


図 5.32: C-SSF1.3 により得られた複素 MLP($J = 12$) の出力 (人工データ 2)

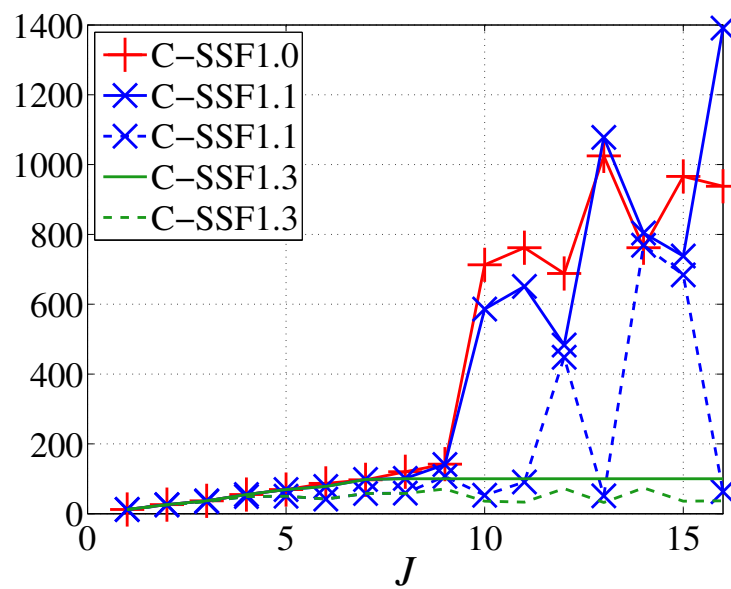


図 5.33: C-SSF1.3 の探索開始数 (実線) と探索枝刈りされなかった探索回数 (点線) (人工データ 2)

第6章 SSF1.4とC-SSF1.3を用いた カオス時系列の予測

実世界には時間と共に変化する複雑な現象が満ちているが、その系の状態遷移規則はより単純な形で記述できる可能性がある。系の状態遷移規則が決定論的であるにも関わらず、複雑な挙動を生み出す現象は決定論的カオスと呼ばれる [9]。決定論的カオスの特徴としてフラクタル性や長期予測不可能性がある。そのため長期的な予測は不可能であるが、短期的な予測はうまくモデルを見つければ精度を向上できる。

本章では決定論的カオス挙動を示すローレンツ方程式 [13] と 2 重振り子 [26] を使い、第 4 章で提案した実 MLP の探索法である SSF1.4 と、第 5 章で提案した複素 MLP の探索法である C-SSF1.3 の性能を評価する計算機実験を行った。SSF1.4 と C-SSF1.3 の特異領域の初期値などの設定は第 4, 5 章と同じとした。

比較する実 MLP の従来法は、2.2 で説明した準 Newton 法の一種の BPQ [22] とした。BPQ の初期重みは区間 $(-1, +1)$ の中からランダムに選択し、各 J で 100 回の試行を行った。比較する複素 MLP の従来法は、3.4 で説明した準 Newton 法の一である C-BFGS [28] とした。C-BFGS の初期重みは実部と虚部共に区間 $(-1, +1)$ の中からランダムに選択し、各 J で 100 回の試行を行った。

複素 MLP は複素数を入力できるが、本章の計算機実験では全て実数を入力した。また、複素 MLP の出力ユニットは実部のみを考慮する式 (3.4) の目的関数を用いて学習を行った。

訓練誤差とテスト誤差は実 MLP と複素 MLP 共に以下の MSE(mean squared error) と

した.

$$MSE = \frac{1}{N} \sum_{\mu=1}^N (\text{Re}(f^\mu) - y^\mu)^2 \quad (6.1)$$

プログラミング言語は MATLAB R2014a を用いた.

6.1 ローレンツ方程式を用いた実験

ローレンツ方程式を以下に示す.

$$\frac{dx}{dt} = \sigma(y - x) \quad (6.2)$$

$$\frac{dy}{dt} = x(\rho - z) - y \quad (6.3)$$

$$\frac{dz}{dt} = xy - \beta z \quad (6.4)$$

ρ, σ, β はそれぞれ 28, 10, 8/3, x, y, z の初期値はそれぞれ 10, -10, 30, Δt は 0.05 とした. 図 6.1 に $t = 0, \Delta t, \dots, 299\Delta t$ の x, y, z の軌道を示す.

ここでは時刻 t の点 $\mathbf{p}_t \leq (x_t, y_t, z_t)^T$ を入力し, $\mathbf{p}_{t+\Delta t}$ を予測するように MLP を学習する問題とした. MLP の出力ユニットは 1 つとし, $x_{t+\Delta t}, y_{t+\Delta t}, z_{t+\Delta t}$ をそれぞれ予測するため 3 つの MLP を用いた. 以下にそれぞれの訓練データを示す. ただしサンプル数 N_{tr} は 300 とした.

$$\equiv \{(\mathbf{p}_{(\mu-1)\Delta t}, x_{\mu\Delta t}), \mu = 1, \dots, N_{tr}\}$$

$$\equiv \{(\mathbf{p}_{(\mu-1)\Delta t}, y_{\mu\Delta t}), \mu = 1, \dots, N_{tr}\}$$

$$\equiv \{(\mathbf{p}_{(\mu-1)\Delta t}, z_{\mu\Delta t}), \mu = 1, \dots, N_{tr}\}$$

テストデータは, 以下に示すように, $t = N_{tr}\Delta t$ 以降とし, サンプル数 N_{test} は 200 とした.

$$\equiv \{(\mathbf{p}_{(N_{tr}+\mu-1)\Delta t}, x_{(N_{tr}+\mu)\Delta t}), \mu = 1, \dots, N_{test}\}$$

$$\equiv \{(\mathbf{p}_{(N_{tr}+\mu-1)\Delta t}, y_{(N_{tr}+\mu)\Delta t}), \mu = 1, \dots, N_{test}\}$$

$$\equiv \{(\mathbf{p}_{(N_{tr}+\mu-1)\Delta t}, z_{(N_{tr}+\mu)\Delta t}), \mu = 1, \dots, N_{test}\}$$

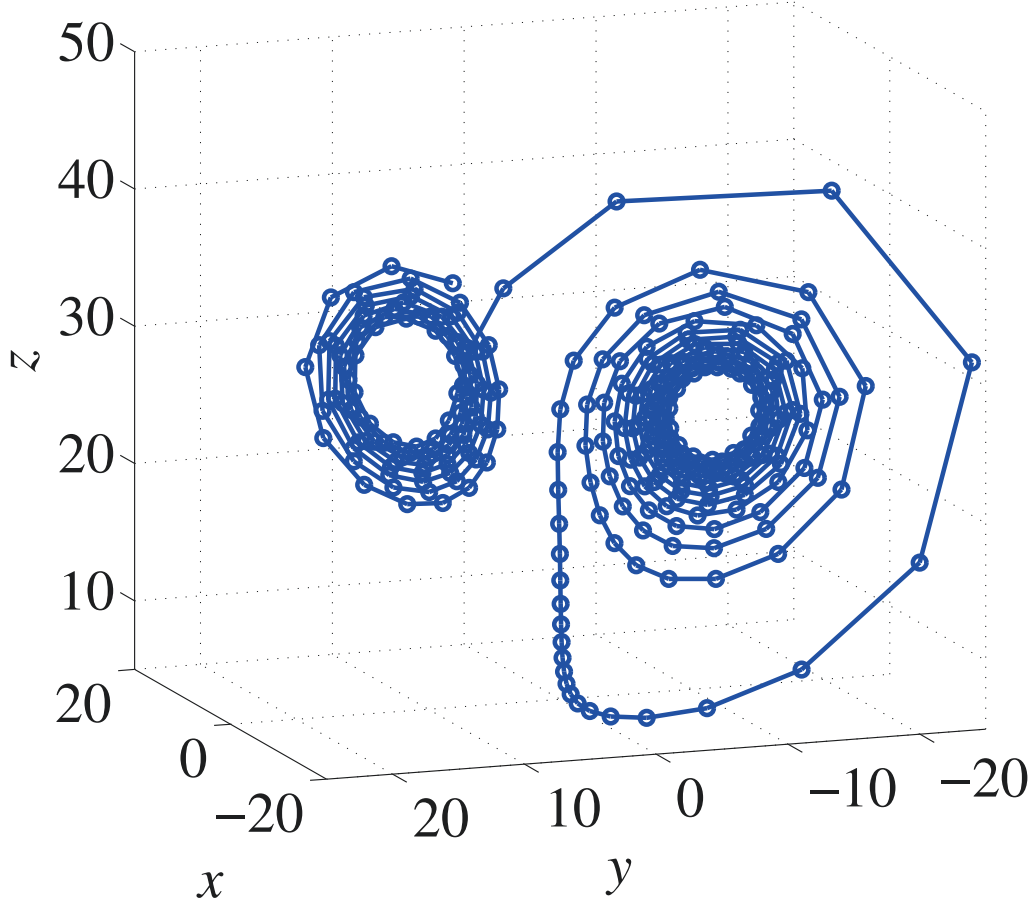


図 6.1: ローレンス方程式から生成された x, y, z の軌道（訓練データ生成領域）

考慮する隠れユニットの最大数は実 MLP は 30, 複素 MLP は 15 とした.

図 6.2 に各 J で得られた最小の訓練誤差を示す. BPQ と C-BFGS では隠れユニットを増やすと逆に訓練誤差が増加することがあった. SSF1.4 と C-SSF1.3 では, 隠れユニット数の増加に伴って訓練誤差が単調減少し, x_t, y_t, z_t に対する最小 MSE はどちらも 10^{-11} 以下となった.

図 6.3 に最小訓練誤差の解のテスト誤差を示す. x_t, y_t, z_t に対するテスト誤差は全て C-SSF1.3 が最小となり, 次いで SSF1.4, BPQ, C-BFGS の順となった.

BPQ, SSF1.4, C-BFGS, C-SSF1.3 の x_t, y_t, z_t に対するテスト誤差が最小の隠れユニット数 (J_x, J_y, J_z) はそれぞれ (4, 9, 15), (15, 26, 27), (7, 5, 7), (12, 9, 11) となった. このときの MLP に $\mathbf{p}_{N_{tr}\Delta t}$ のみを与えて $\mathbf{p}_{(N_{tr}+1)\Delta t}, \mathbf{p}_{(N_{tr}+2)\Delta t}, \dots, \mathbf{p}_{(N_{tr}+N_{test})\Delta t}$ を順にフリーラン予測した. フリーラン予測では, $t = (N_{tr} + 1)\Delta t$ 以降の正しい $\mathbf{p}_t$ は与えず, それらに対応する MLP の予測値を MLP の入力として利用した. 図 6.4, 6.5, 6.6, 6.7 に, BPQ, SSF1.4,

C-BFGS, C-SSF1.3 のフリーラン予測の結果を示す。このときの BPQ, SSF1.4, C-BFGS, C-SSF1.3 の MSE はそれぞれ 238.82, 0.28, 283.78, 0.01 となった。BPQ と C-BFGS は予測が大幅にずれていることがわかる。SSF1.4 と C-SSF1.3 はうまく予測できたが、C-SSF1.3 の予測の方が正確である。

図 6.8, 6.9 に各 J での処理時間と重み更新平均回数を示す。BPQ, SSF1.4, C-BFGS, C-SSF1.3 の合計時間はそれぞれ 13.1, 1.13, 1.51, 10.0 時間となり、SSF1.4 が最も早く、BPQ が最も時間が掛かった。BPQ, C-BFGS, C-SSF1.3 は隠れユニットの増加に伴って処理時間が増加する傾向があったが、SSF1.4 は隠れユニット数が多いとき処理時間が減少した。これは隠れユニットを増やしても訓練誤差がそれ以上減少しないことがあったためと考えられる。また、C-BFGS の重み更新平均回数は C-SSF1.3 よりも少ない傾向があった。これは、C-BFGS では初期値付近の比較的訓練誤差とテスト誤差が大きい解に収束してしまうことが多かったためと考えられる。また、C-SSF1.3 では SSF1.4 よりもテスト誤差が小さい解を得たが、C-BFGS のテスト誤差は BPQ よりも大きかった。この原因としては、複素 MLP の探索空間の方が実 MLP よりも探索が困難な形状をしていたためと考えられる。

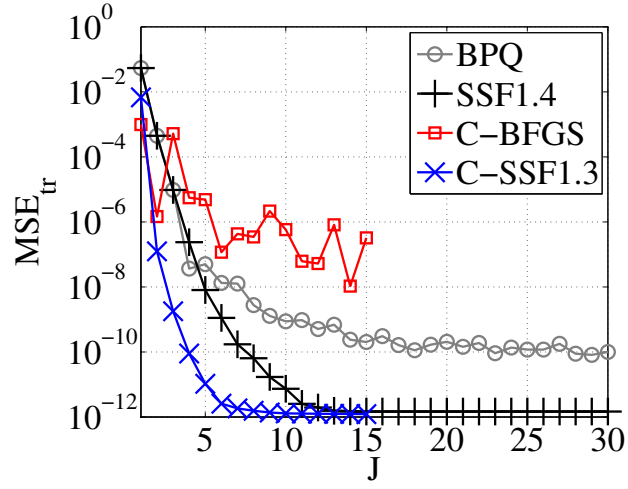
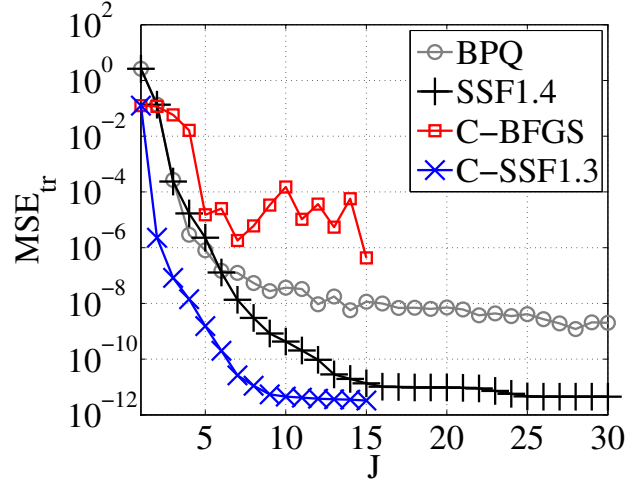
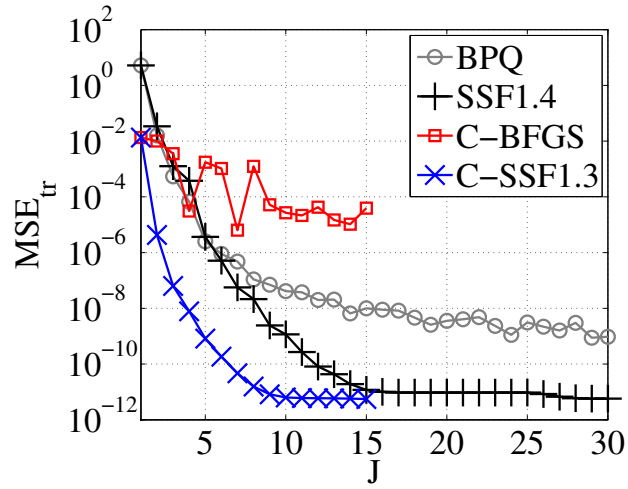
(a) x_t (b) y_t (c) z_t

図 6.2: 訓練誤差 (ローレンツ方程式)

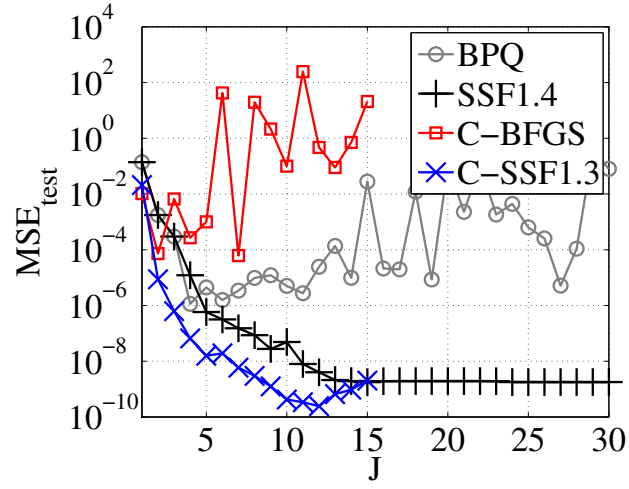
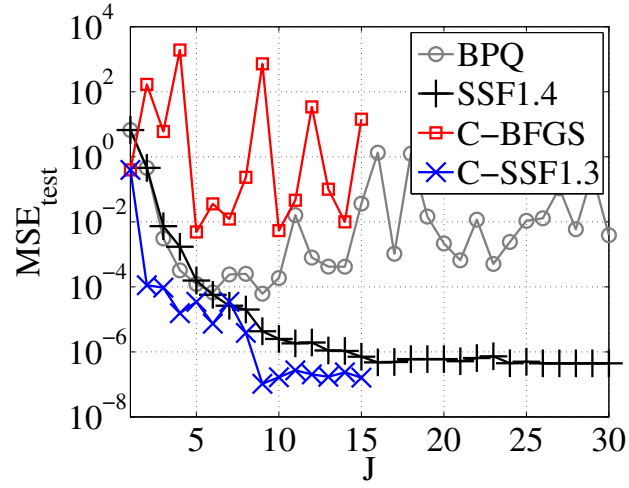
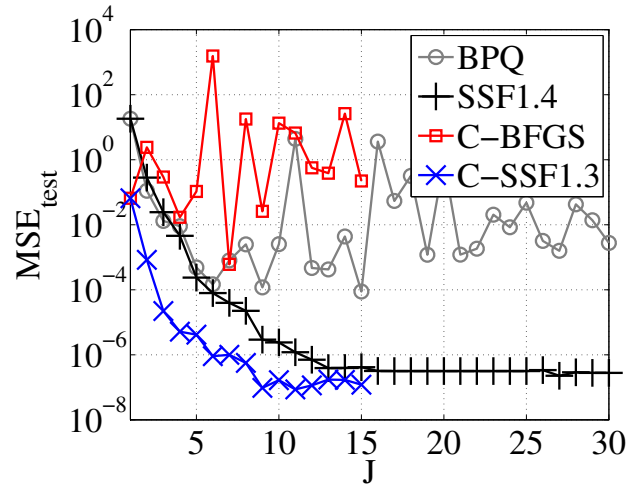
(a) x_t (b) y_t (c) z_t

図 6.3: テスト誤差 (ローレンツ方程式)

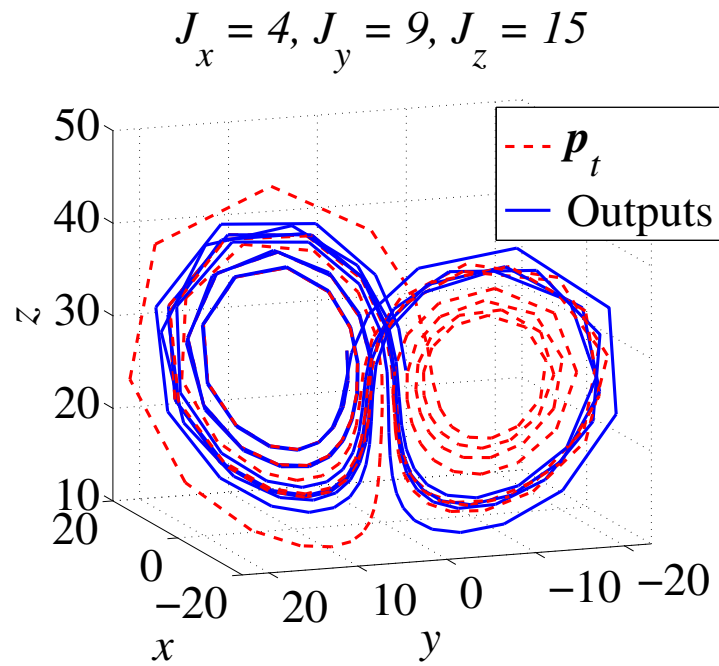


図 6.4: BPQ のフリーラン予測 (ローレンツ方程式)

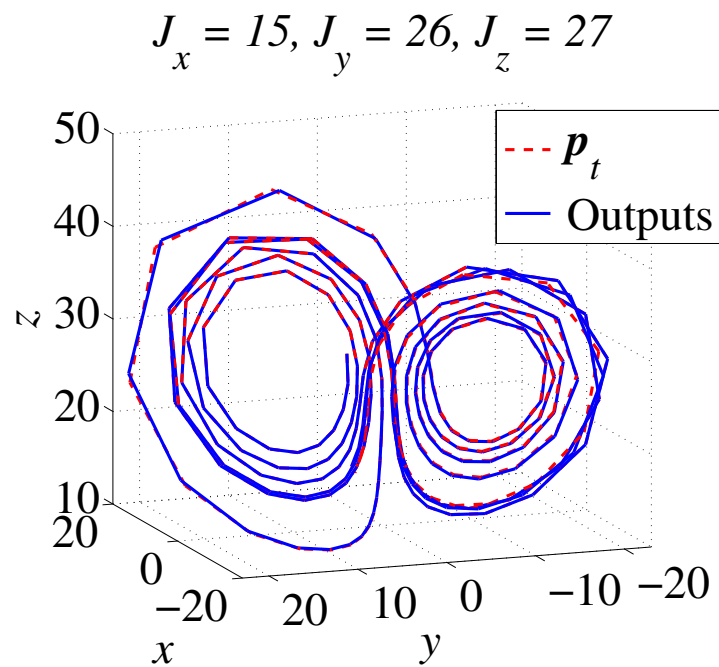


図 6.5: SSF1.4 のフリーラン予測 (ローレンツ方程式)

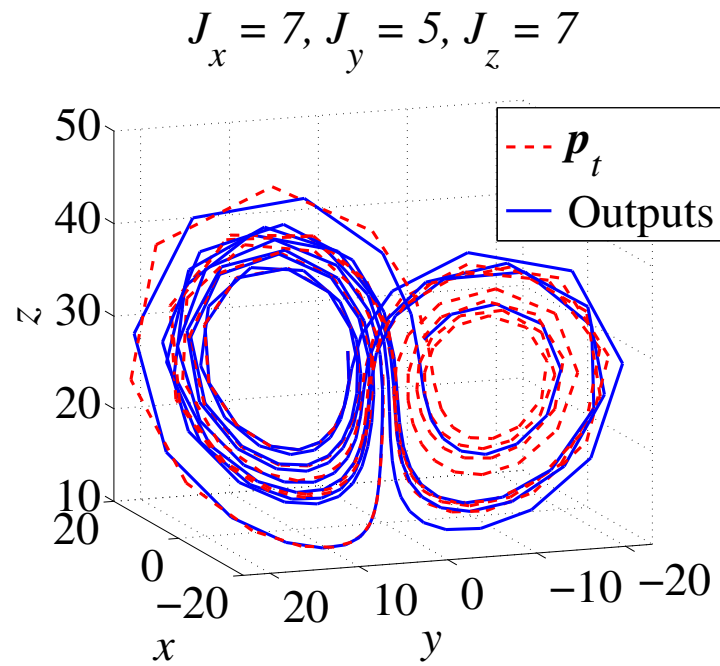


図 6.6: C-BFGS のフリーラン予測 (ローレンツ方程式)

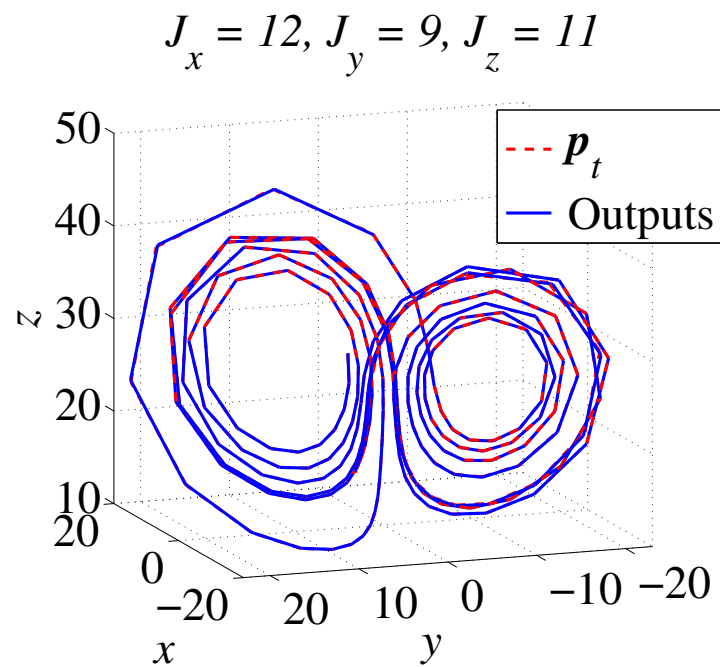


図 6.7: C-SSF1.3 のフリーラン予測 (ローレンツ方程式)

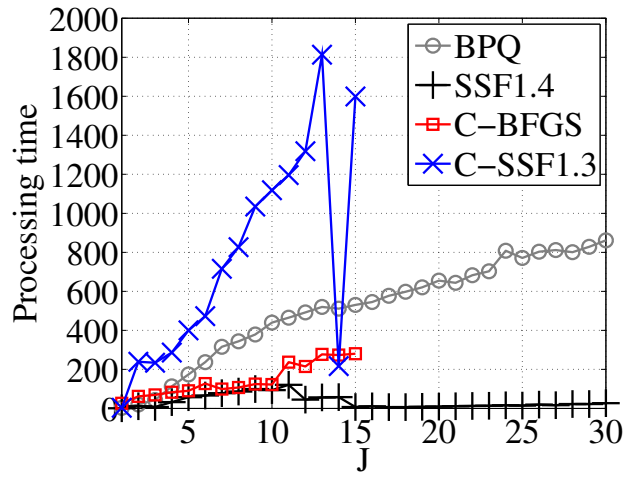
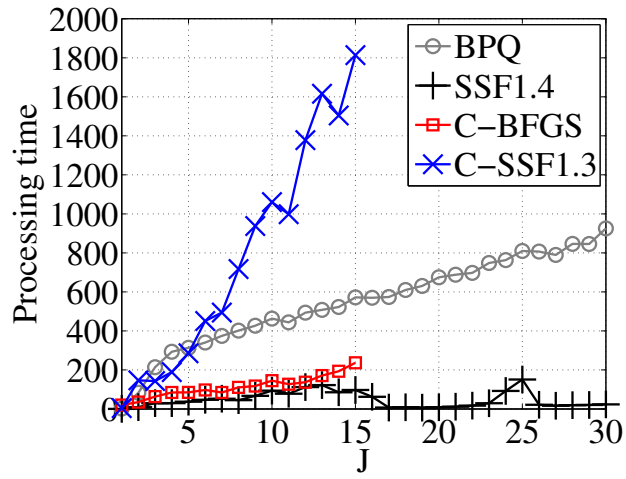
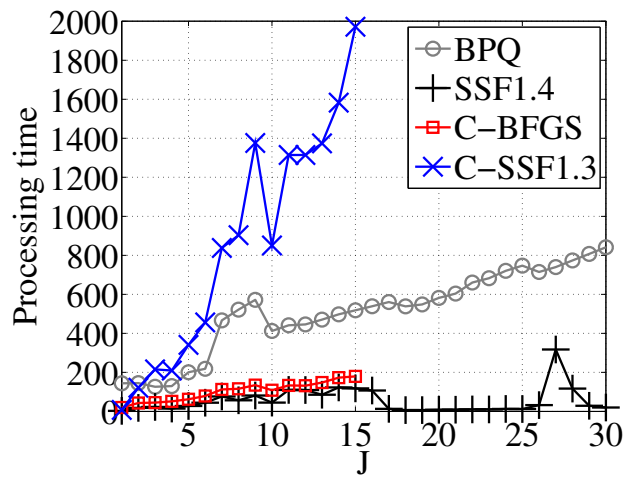
(a) x_t (b) y_t (c) z_t

図 6.8: 処理時間 (秒) (ローレンツ方程式)

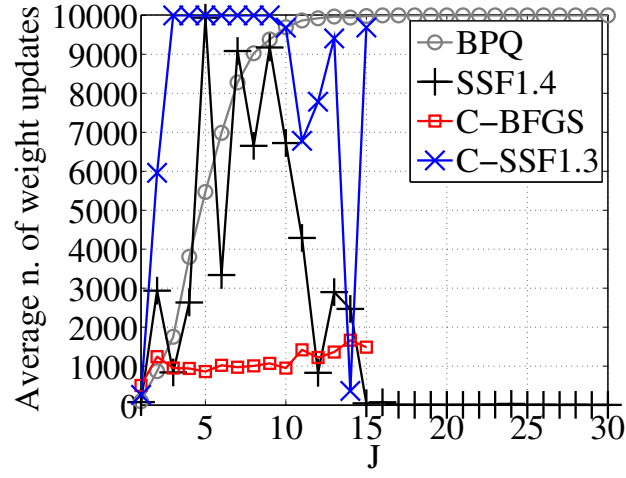
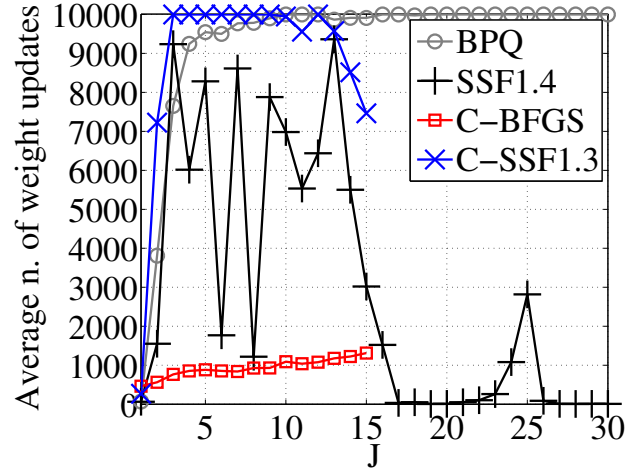
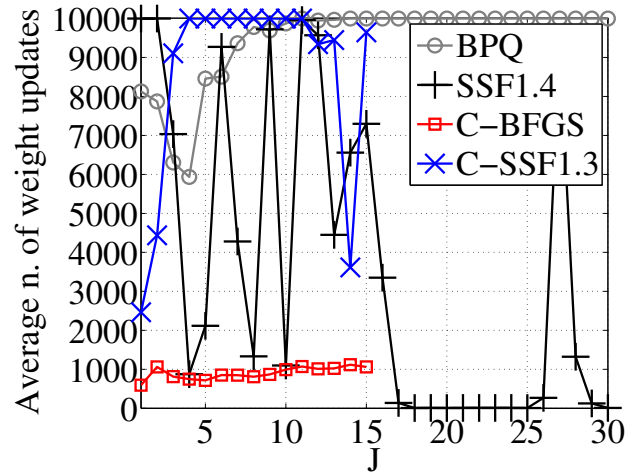
(a) x_t (b) y_t (c) z_t

図 6.9: 重み更新平均回数 (ローレンツ方程式)

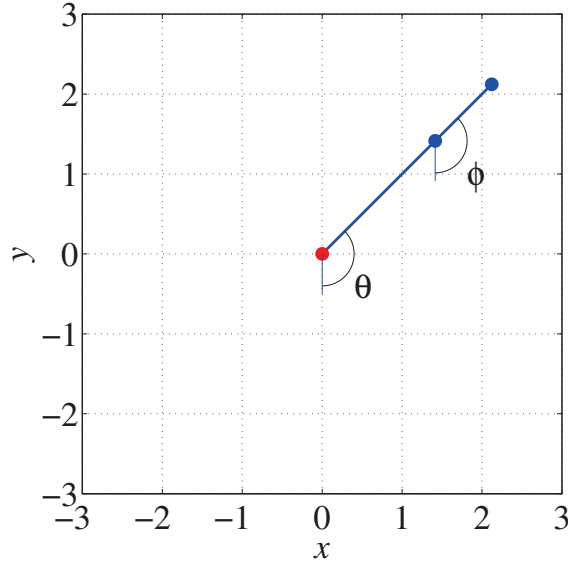


図 6.10: 2重振り子の初期位置

6.2 2重振り子を用いた実験

図 6.10 に示す 2 重振り子を用いて実験を行った．ここで一つ目の棒の始点は $(0, 0)$ ，棒の長さは l_1 ，棒の先端の重りの重さは m_1 とし，棒自体の質量は 0 とした．棒の角度 θ は真下を向いているときを 0° とした．二つ目の棒の始点の一つ目の棒の先端とし，長さは l_2 ，先端の重りは m_2 とした．棒の角度 ϕ は一つ目と同様，真下を向いているときを 0° とした．

この 2 重振り子のラグランジュ関数 L は以下である．ただし， T は運動エネルギー， U はポテンシャルエネルギーとし， $\dot{\theta} = \frac{d\theta}{dt}$, $\dot{\phi} = \frac{d\phi}{dt}$, $M = m_1 + m_2$, $\psi = \theta - \phi$ とする．

$$L = T - U, \quad (6.5)$$

$$T = \frac{1}{2} M l_1^2 \dot{\theta}^2 + \frac{1}{2} m_2 l_2^2 \dot{\phi}^2 + m_2 l_1 l_2 \dot{\theta} \dot{\phi} \cos \psi, \quad (6.6)$$

$$U = -M g l_1 \cos \theta - m_2 g l_2 \cos \phi \quad (6.7)$$

ラグランジュの方程式は以下である．

$$\left(\frac{d}{dt} \right) \frac{\partial L}{\partial \dot{\theta}_1} \left(\frac{\partial L}{\partial \theta_1} = 0, \quad \left(\frac{d}{dt} \right) \frac{\partial L}{\partial \dot{\theta}_2} \left(\frac{\partial L}{\partial \theta_2} = 0 \right. \right. \quad (6.8)$$

上記式を $\ddot{\theta}(\leq \frac{d^2\theta}{dt^2}), \ddot{\phi}(\leq \frac{d^2\phi}{dt^2})$ について解くと以下を得る.

$$\ddot{\theta} = \frac{g(m_2 \sin \phi \cos \psi - M \sin \theta) - m_2(l_1 \dot{\theta}^2 \cos \psi + l_2 \dot{\phi}^2) \sin \psi}{l_1(M - m_2 \cos^2 \psi)} \quad (6.9)$$

$$\ddot{\phi} = \frac{Mg(\sin \theta \cos \psi - \sin \phi) + (m_2 l_2 \dot{\phi}^2 \cos \psi + M l_1 \dot{\theta}^2) \sin \psi}{l_2(M - m_2 \cos^2 \psi)} \quad (6.10)$$

m_1, m_2, l_1, l_2, g はそれぞれ 1, 1, 2, 1, 9.8 とし, $\theta, \phi, \dot{\theta}, \dot{\phi}$ の初期値は $3/4\pi, 3/4\pi, 2, 2, \Delta t$ は 0.05 とした.

ここでは以下を入力とし, $\theta_{t+\Delta t}, \phi_{t+\Delta t}$ を予測するように MLP を学習する問題とした.

$$\mathbf{q}_t \leq (\theta_{t-9\Delta t}, \theta_{t-8\Delta t}, \dots, \theta_t, \phi_{t-9\Delta t}, \phi_{t-8\Delta t}, \dots, \phi_t)^T$$

6.1 と同様, MLP の出力ユニットは 1 つとし, $\theta_{t+\Delta t}, \phi_{t+\Delta t}$ をそれぞれ予測するため 2 つの MLP を用いた. 以下にそれぞれの訓練データを示す. サンプル数 N_{tr} は 10,000 とした.

$$\equiv \{(\mathbf{q}_{(8+\mu)\Delta t}, \theta_{(9+\mu)\Delta t}), \mu = 1, \dots, N_{tr} |$$

$$\equiv \{(\mathbf{q}_{(8+\mu)\Delta t}, \phi_{(9+\mu)\Delta t}), \mu = 1, \dots, N_{tr} |$$

テストデータは, 以下に示すように, $t = 20,000\Delta t$ 以降とし, サンプル数 N_{test} は 100 とした.

$$\equiv \{(\mathbf{q}_{(20,008+\mu)\Delta t}, \theta_{(20,009+\mu)\Delta t}), \mu = 1, \dots, N_{test} |$$

$$\equiv \{(\mathbf{q}_{(20,008+\mu)\Delta t}, \phi_{(20,009+\mu)\Delta t}), \mu = 1, \dots, N_{test} |$$

考慮する隠れユニットの最大数は複素 MLP は 10, 実 MLP は 20 とした.

図 6.11 に各 J で得られた最小の訓練誤差を示す. 図の縦軸は対数表示とした. C-BFGS は常に MSE が最大であった. これは, ローレンツ方程式の実験と同様, 複素 MLP の探索空間の方が実 MLP よりも探索が困難な形状をしていたためと考えられる. 一方, C-SSF1.3 は BPQ や SSF1.4 よりも小さい訓練誤差の解を得た.

図 6.12 に最小訓練誤差の解のテスト誤差を示す. 6.1 の結果と同様, θ_t, ϕ_t に対するテスト誤差はどちらにおいても C-SSF1.3 が最小となり, 次いで SSF1.4, BPQ, C-BFGS の順となった.

BPQ, SSF1.4, C-BFGS, C-SSF1.3 の θ_t に対するテスト誤差が最小となるモデルのテストデータに対する出力を図 6.13, 6.14, 6.15, 6.16 に, ϕ_t に対するテスト誤差が最小となる

モデルのテストデータに対する出力を図 6.17, 6.18, 6.19, 6.20 に示す. ただし, 以下のよう
に角度 θ_t, ϕ_t を (x, y) 平面に変換した値 $z_t^{(1)}, z_t^{(2)}$ を表示した.

$$z_t^{(1)} = x_t^{(1)} + iy_t^{(1)}, \quad x_t^{(1)} = l_t^{(1)} \sin \theta_t, \quad y_t^{(1)} = l_t^{(1)} \cos \theta_t, \quad (6.11)$$

$$z_t^{(2)} = x_t^{(2)} + iy_t^{(2)}, \quad x_t^{(2)} = x_t^{(1)} + l_t^{(2)} \sin \phi_t, \quad y_t^{(2)} = y_t^{(1)} + l_t^{(2)} \cos \phi_t \quad (6.12)$$

θ_t の予測結果はどの探索法でもほとんど変わらないが, ϕ_t の予測結果は C-SSF1.3 が
最も精度が良く, 次いで SSF1.4 がよく予測できていることがわかる.

図 6.21, 6.22 に各 J での処理時間と重み更新平均回数を示す. ローレンツ方程式を用
いた計算機実験とは異なり, SSF1.4 も他の探索法と同様, 隠れユニットの増加に伴って
処理時間が増加する傾向があった. BPQ, SSF1.4, C-BFGS, C-SSF1.3 の合計時間はそれ
ぞれ 67.8, 14.1, 4.62, 16.1 時間となり, C-BFGS が最も早く, BPQ が最も時間がかかっ
た. C-BFGS が最も早かった原因としては, ローレンツ方程式を用いた計算機実験と同
様, C-BFGS の重み更新平均回数は C-SSF1.3 よりも少ない傾向があり, C-BFGS では初
期値付近の比較的訓練誤差とテスト誤差が大きい解に収束してしまう傾向があったためと
考えられる.

BPQ, SSF1.4, C-BFGS, C-SSF1.3 の θ_t, ϕ_t に対するテスト誤差が最小の隠れユニット数
(J_θ, J_ϕ) はそれぞれ (18, 1), (20, 18), (10, 10), (8, 9) となった. このときの MLP に $\mathbf{q}_{20,009\Delta t}$ の
みを与えて $\theta_{20,010\Delta t}, \theta_{20,011\Delta t}, \times\times, \theta_{20,019\Delta t}, \phi_{20,010\Delta t}, \phi_{20,011\Delta t}, \times\times, \phi_{20,019\Delta t}$ を順にフリー
ラン予測した. フリーラン予測では, $t = 20,010\Delta t$ 以降の正しい θ_t, ϕ_t は与えず, それら
に対応する MLP の出力を MLP の入力として利用した. 図 6.23, 6.24, 6.25, 6.26 に, BPQ,
SSF1.4, C-BFGS, C-SSF1.3 のフリーラン予測の結果を示す. 図から C-SSF1.3 のみがう
まく予測できていることがわかる.

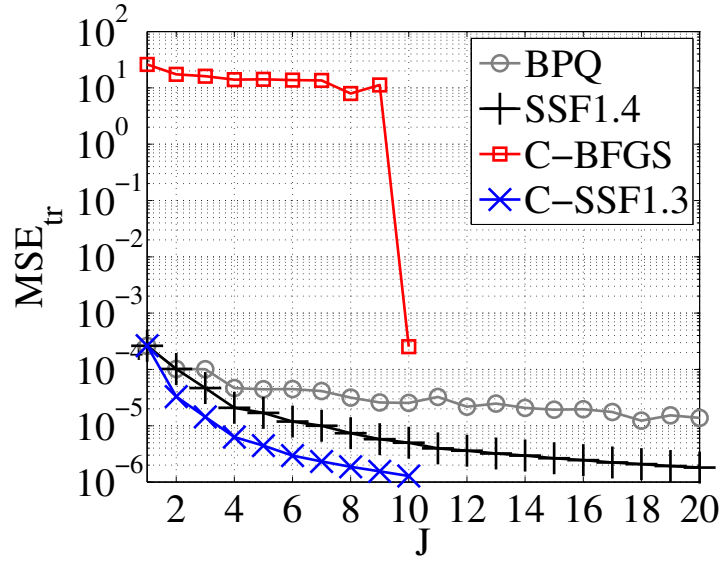
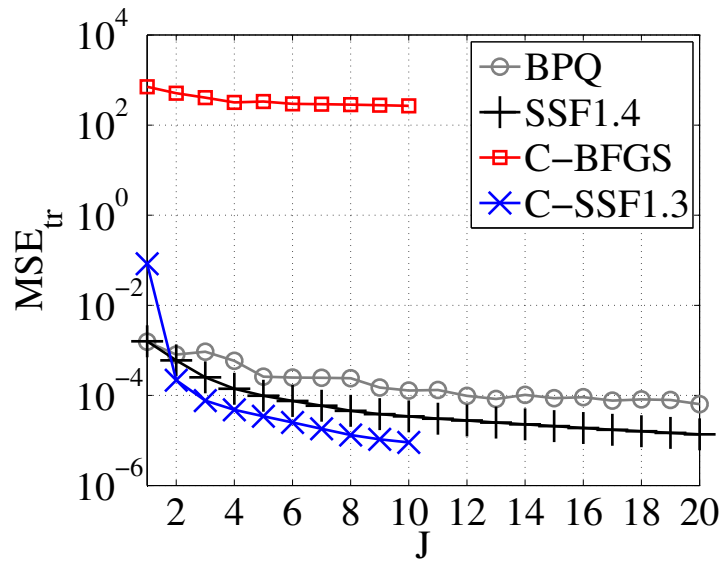
(a) θ_t (b) ϕ_t

図 6.11: 訓練誤差 (2重振り子)

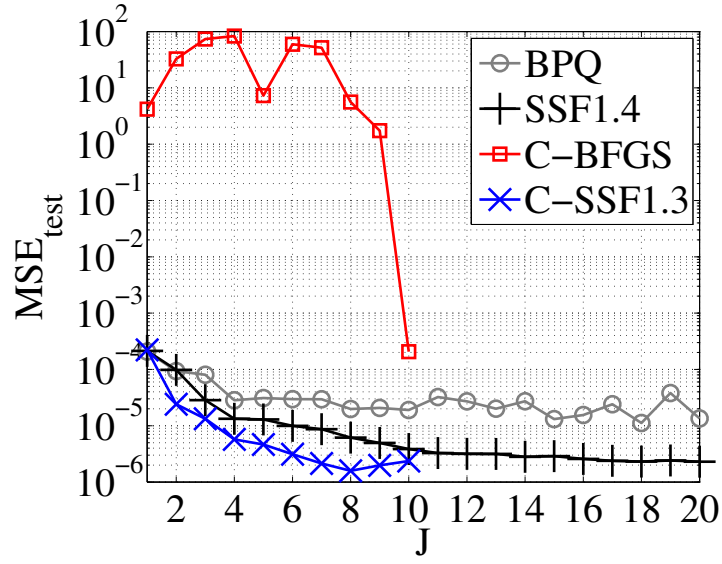
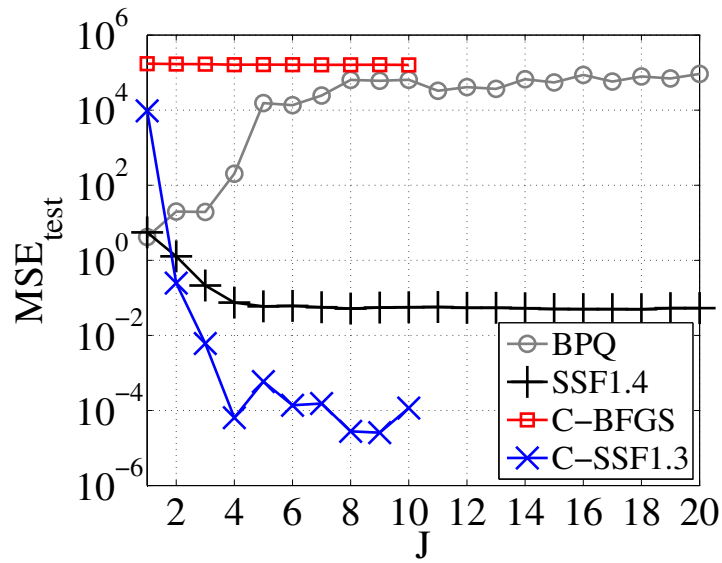
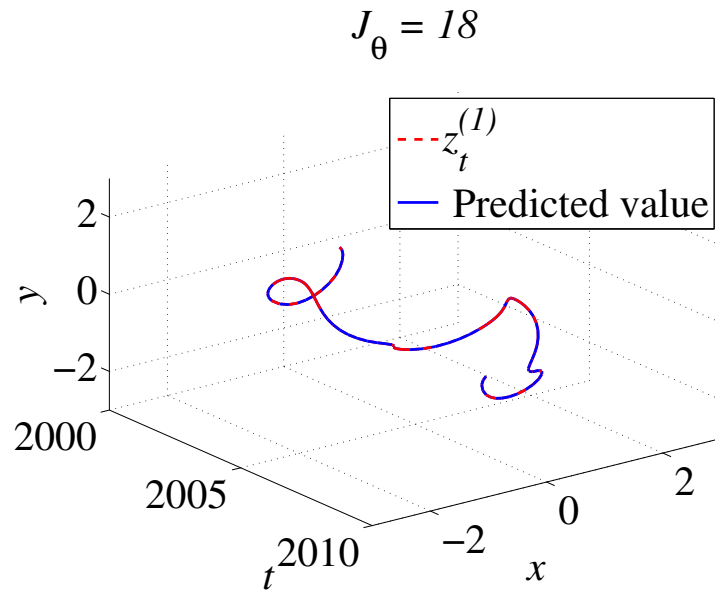
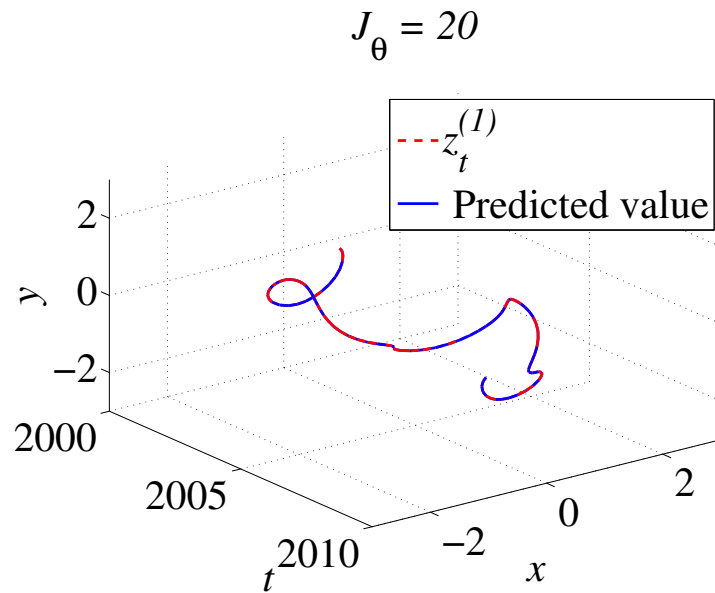
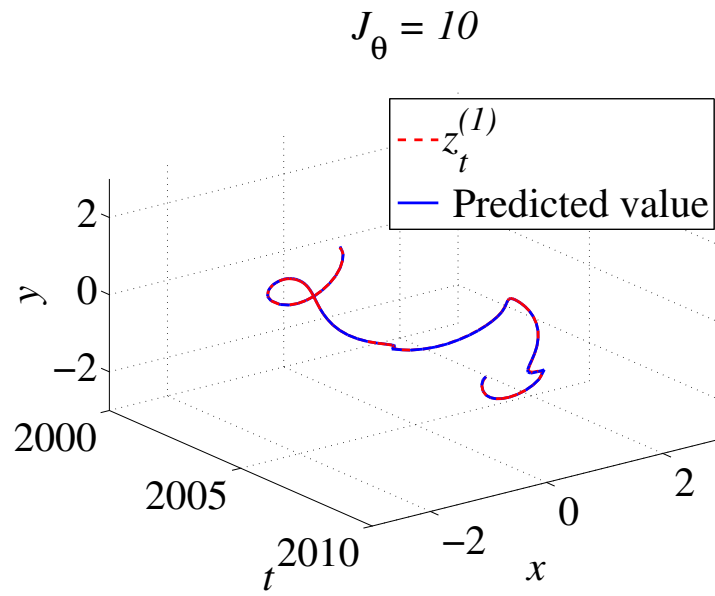
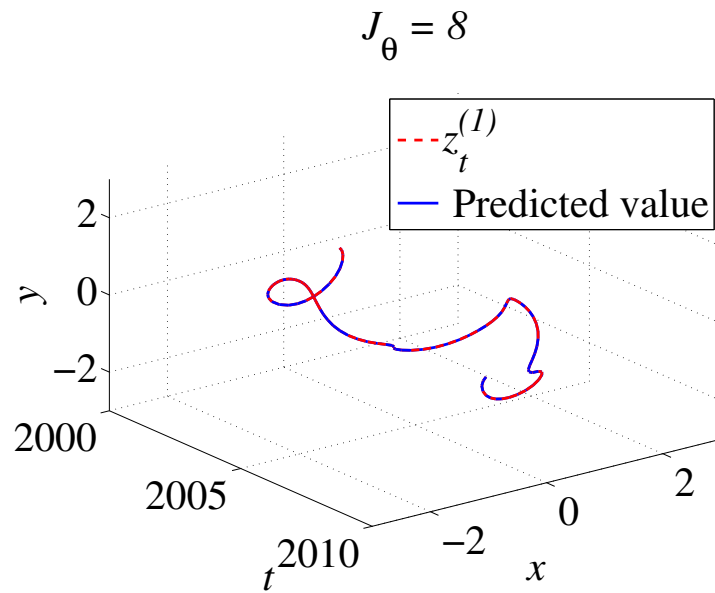
(a) θ_t (b) ϕ_t

図 6.12: テスト誤差 (2重振りをういた実験)

図 6.13: BPQ の 2 重振り子の θ のテストデータ予測結果図 6.14: SSF1.4 の 2 重振り子の θ のテストデータ予測結果

図 6.15: C-BFGS の 2 重振り子の θ のテストデータ予測結果図 6.16: C-SSF1.3 の 2 重振り子の θ のテストデータ予測結果

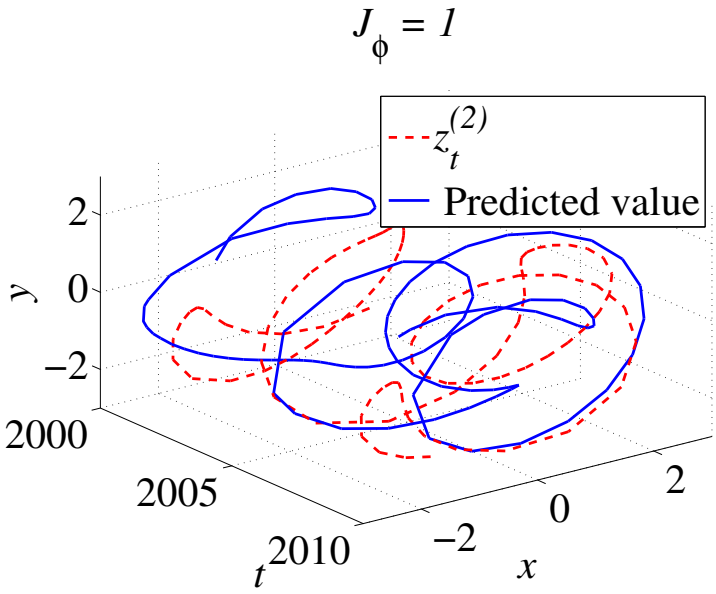


図 6.17: BPQ の 2 重振り子の θ のテストデータ予測結果

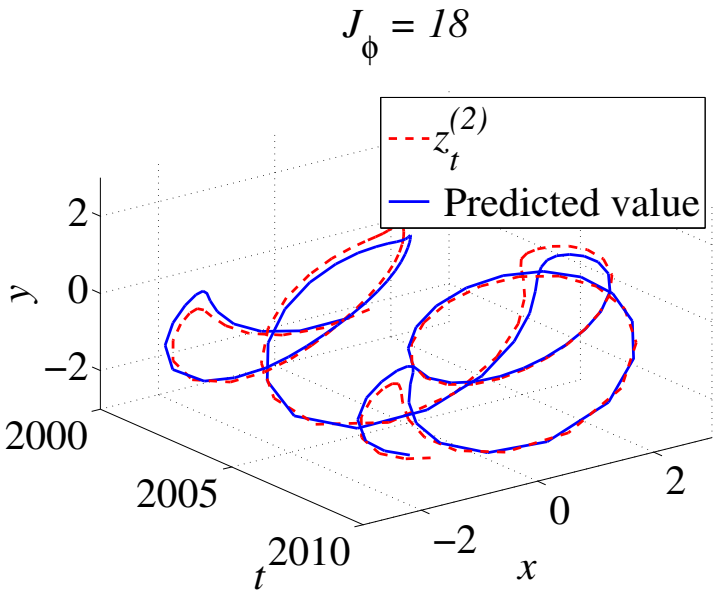
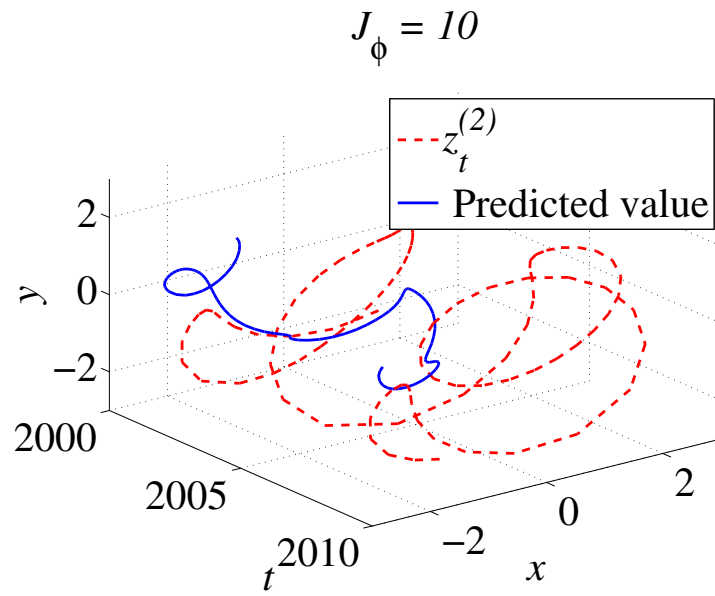
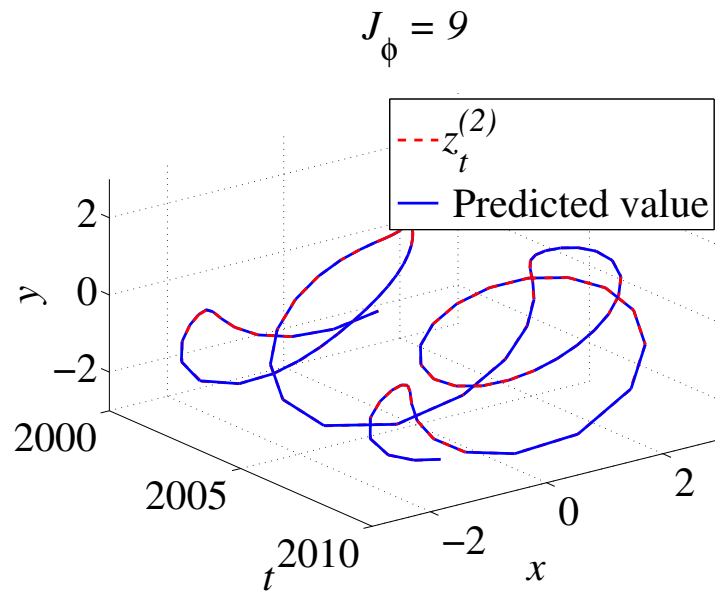


図 6.18: SSF1.4 の 2 重振り子の θ のテストデータ予測結果

図 6.19: C-BFGS の 2 重振り子の θ のテストデータ予測結果図 6.20: C-SSF1.3 の 2 重振り子の θ のテストデータ予測結果

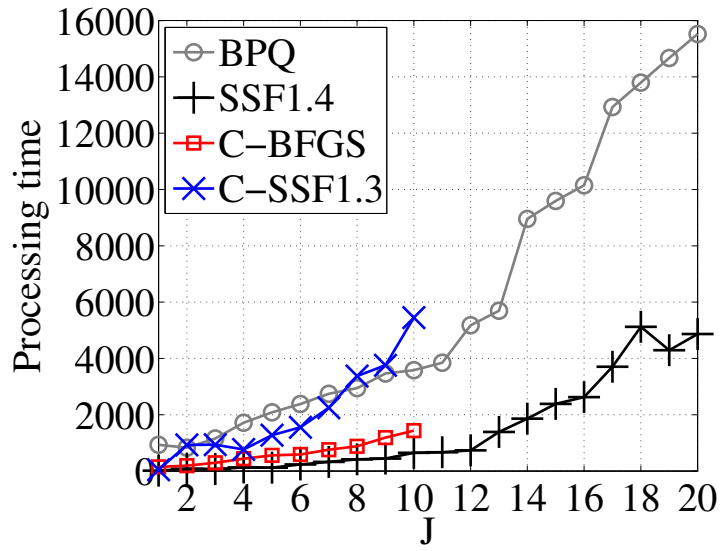
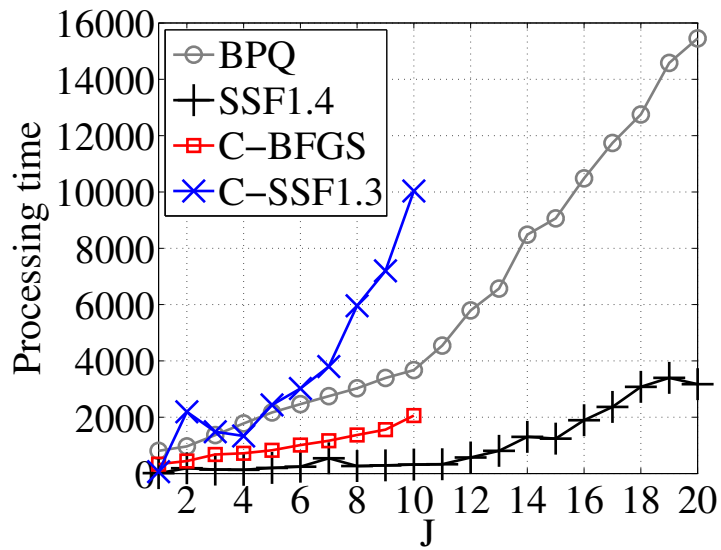
(a) θ_t (b) ϕ_t

図 6.21: 処理時間 (秒) (2重振り子)

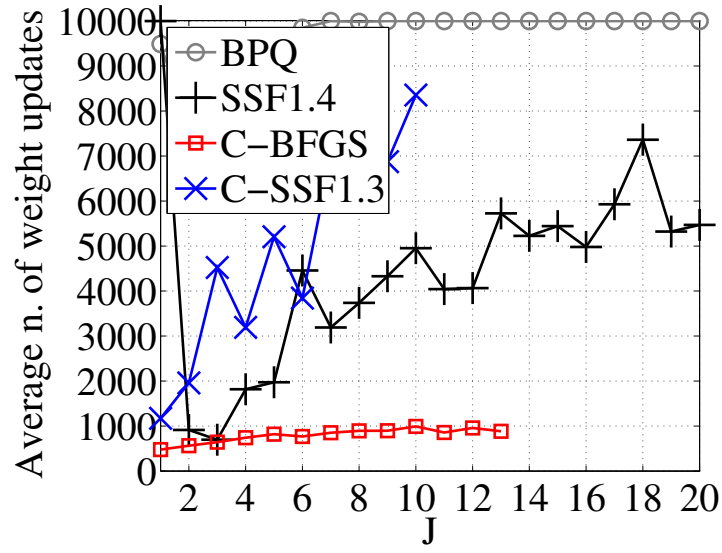
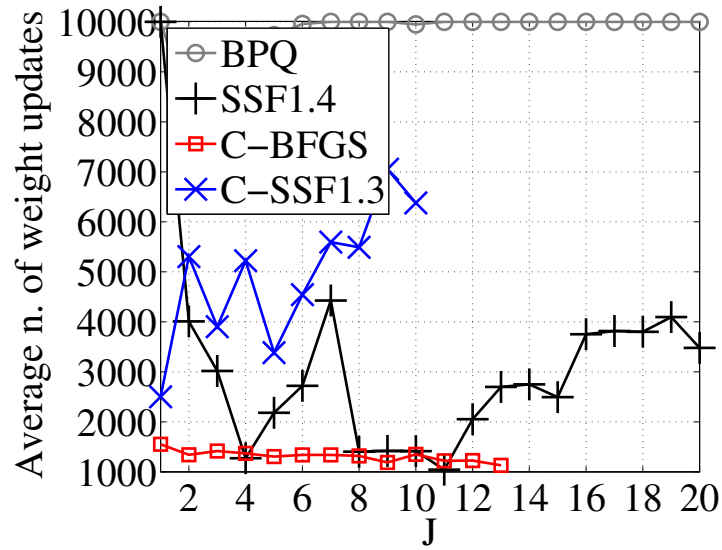
(a) θ_t (b) ϕ_t

図 6.22: 重み更新平均回数 (2重振りをういた実験)

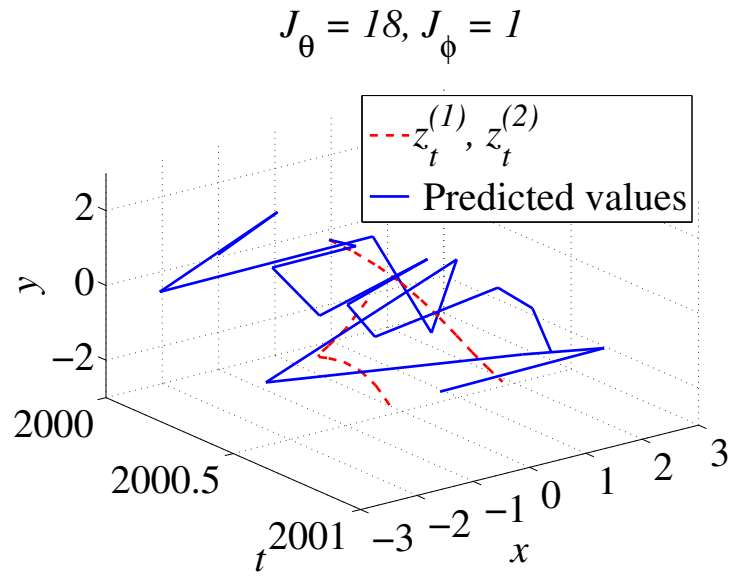


図 6.23: BPQ のフリーラン予測 (2重振り子)

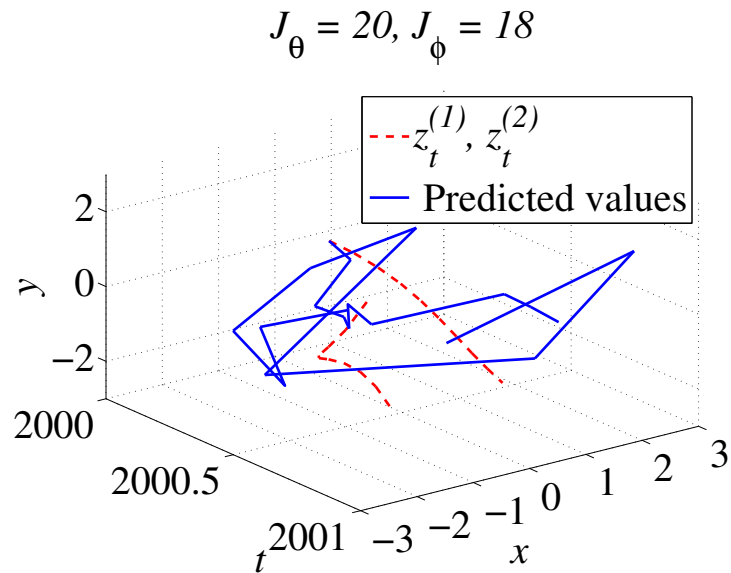


図 6.24: SSF1.4 のフリーラン予測 (2重振り子)

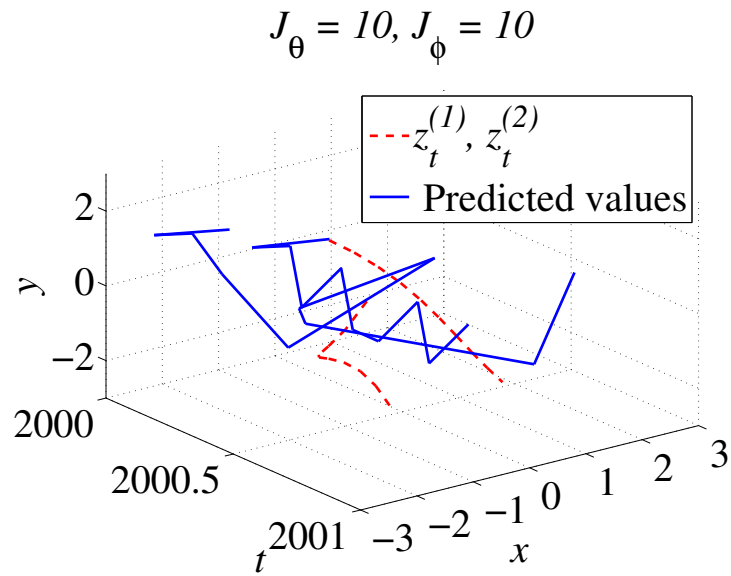


図 6.25: C-BFGS のフリーラン予測 (2重振り子)

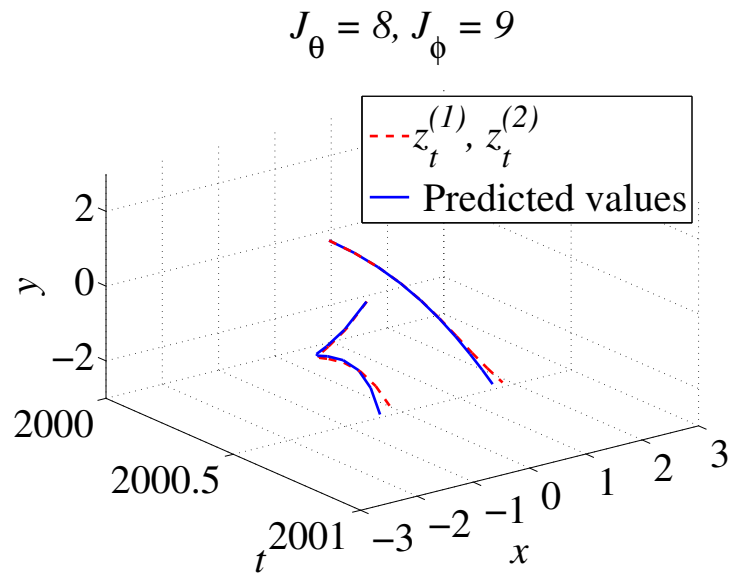


図 6.26: C-SSF1.3 のフリーラン予測 (2重振り子)

6.3 まとめ

本章では，生成される挙動が決定論的カオスとされるローレンツ方程式と2重振り子のモデルを用いて，SSF1.4とC-SSF1.3の性能を評価した．どちらの実験においても準Newton法の一種のBPQとC-BFGSでは隠れユニットを増加しても訓練誤差が減少しないことがあり，良質の解が得られなかったが，SSF1.4とC-SSF1.3では訓練誤差が単調減少し，ローレンツ方程式の実験では $200\Delta t$ 先までうまくフリーラン予測できた．2重振り子の実験の Δt 先を予測するテストではSSF1.4は少しずれているが，C-SSF1.3ではうまく予測できた．また， $10\Delta t$ 先までのフリーラン予測においてもC-SSF1.3はうまく予測できた．

今後の課題としてはカオス時系列のフリーラン予測の精度をより長期に渡って向上することが挙げられる．

第7章 結論と展望

本論文では，高速かつ安定して良質の解を得るため，学習が停滞する特異領域を逆に利用して探索する実 MLP の探索法と複素 MLP の探索法を提案した．

7.1 結論

第2章では，実 MLP の従来の探索法である BP や BPQ などについて述べ，局所最適解や特異領域の問題があることについて述べた．

第3章では，複素 MLP の従来法である C-BP や C-BFGS についてや，Wirtinger 微分について述べ，実 MLP と同様，局所最適解や特異領域の問題があることについて述べた．

第4章では，学習が停滞する特異領域を逆に利用して実 MLP の解を探索する SSF1.2 を提案した．SSF1.2 は実 MLP($J-1$) の探索で得られた解に可約性写像を適用して形成した特異領域から実 MLP(J) の探索を行うため，訓練誤差を単調に減少できる特長があり，初期値をランダムに設定して BP や BPQ を用いて 100 回探索する方法よりも良質の解を得ることができた．しかし SSF1.2 では，隠れユニットの増加に伴って特異領域が増えることで処理時間が大幅に増加した．この問題を解決するため，現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を導入した SSF1.3 を提案した．また，特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し，探索数の上限を設定する手法により一層高速化した SSF1.4 を提案した．SSF1.4 は SSF1.2 の解品質を劣化させることなく高速化でき，人工データを用いた実験では BPQ の 4 倍以上，実データを用いた実験では BPQ の 2 倍以上早く処理が終了した．また，BPQ では隠れユニットが多

いほど訓練誤差が単調減少しない傾向が強くなったが、SSF1.4 では訓練誤差を単調減少させ、BPQ よりも小さいテスト誤差の解を得た。

第5章では、SSF1.2 を複素 MLP に拡張した C-SSF1.0 を提案した。C-SSF1.0 は複素 $\text{MLP}(J-1)$ の探索で得られた解に可約性写像を適用して形成した特異領域から複素 $\text{MLP}(J)$ の探索を行うことにより、SSF1.2 と同様、訓練誤差を単調に減少させ、初期値をランダムに設定して C-BP や C-BFGS を用いて 100 回探索する方法よりも良質の解を得ることができた。しかし C-SSF1.0 では、隠れユニットの増加に伴って特異領域が増えることで処理時間が大幅に増加したため、SSF1.3 と同様、現在の探索が以前の探索経路と合流する場合は現在の探索を枝刈りする手法を導入した C-SSF1.1 を提案した。更に、C-SSF1.1 をより高速化するため、SSF1.4 と同様、特異領域上の Hesse 行列の固有値を基に探索の優先順位を決定し、探索数の上限を設定する手法により一層高速化した C-SSF1.3 を提案した。C-SSF1.3 は各隠れユニット毎に実行する探索数に上限を設定し、特異領域上の Hesse 行列の固有値が小さい順に探索を行うことにより、解品質を劣化させることなく高速化でき、人工データ 1 を用いた実験では C-BFGS よりも速く処理が終了し、人工データ 2 を用いた実験では C-BFGS の 2 倍以上早く処理が終了した。また、BPQ と同様、C-BFGS では隠れユニットが多いほど訓練誤差が単調減少しない傾向が強くなったが、C-SSF1.3 では訓練誤差を単調減少させ、C-BFGS よりも小さいテスト誤差の解が得られた。

第6章では、生成される挙動が決定論的カオスとされるローレンツ方程式と 2 重振り子のモデルを用いて、SSF1.4 と C-SSF1.3 の性能を評価した。第4, 5章と同様、どちらの実験においても BPQ と C-BFGS では隠れユニットを増加しても訓練誤差が減少しないことがあり、良質の解が得られなかったが、SSF1.4 と C-SSF1.3 では訓練誤差が単調減少し、良質の解が得られた。また、どちらの実験においても 1 ステップ先を予測するように MLP を学習したが、SSF1.4 と C-SSF1.3 ではローレンツ方程式の実験では 1 ステップ先を予測する MLP を用いて 200 ステップ先まで順に予測できたが、C-SSF1.3 の方が正確であった。2 重振り子の実験の 1 ステップ先を予測するテストでは SSF1.4 は少しずれているが、C-SSF1.3 ではうまく予測できた。また、C-SSF1.3 のみ 10 ステップ先までうまく予測できた。

7.2 展望

今後の展望としては、より多様なデータを用いて提案法を評価することや、提案法のノイズ耐性を調べる事が挙げられる。また、今回の計算機実験では正則化手法を用いなかったが、ノイズがある場合は正則化手法が有効に働く可能性があるため、正則化手法を用いて実験することも挙げられる。ただし、正則化を用いる場合はパラメータ空間が変形することに留意が必要である。

また、特異領域や特異点をパラメータ空間に持つ学習モデルはMLPに限らず、ガウス混合モデルや隠れマルコフモデルなどの有用な学習モデルの多くに共通していることに着目し、特異モデルやその学習に関する数学的解明がなされて来た [29], [30]. そのため解析結果を基に、SSFのアイデアを上記の学習モデルに応用することにより、これらの特異モデルにおいても高速に安定して良質の解が得られるようになる可能性がある。

さらに、特異モデルのモデル選択においても、従来のサンプリング法 (MCMC(Markov chain Monte Carlo) など) では実行速度が極端に遅いが、SSFのように探索することで、特異モデルにおける高速なサンプリングが可能となることが期待できる。

謝 辞

筆者が中部大学大学院 工学研究科情報工学専攻 後期博士課程在学中に，本研究を進めるにあたり，多くの方々のご支援とご協力頂きました．ここに感謝の意を表すとともにお礼申し上げます．研究の遂行にあたり，常日頃ご指導頂いた同専攻 中野良平教授に深く感謝の意を表します．本論文の副査として，有益なご討論，ご助言頂いた同専攻 岩堀祐之教授，山内康一郎教授，静岡県立大学 経営情報学部 斉藤和巳教授に深謝の意を表します．また，十分な金銭的支援を頂いた日本学術振興会に深謝の意を表します．最後に，日頃からご討論，ご助言頂いた中野研究室の皆様に感謝の意を表します．

参考文献

- [1] S. Amari, “Natural Gradient Works Efficiently in Learning,” *Neural Comput.*, Vol.10, No.2, pp.251-276, 1998.
- [2] S. Amari, H. Park, and K. Fukumizu, “Adaptive method of realizing natural gradient learning for multilayer perceptrons,” *Neural Comput.*, Vol.12, No.6, pp.1399-1409, 2000.
- [3] K.K. Delgado, “The complex gradient operator and the CR-calculus,” *ECE275A-Lecture Supplement*, 2006.
- [4] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern classification* (2nd edition), John Wiley & Sons, Inc., New York, 2001.
- [5] K. Fukumizu, and S. Amari, “Local minima and plateaus in heirarchical structure of multilayer perceptrons,” *Neural Networks*, Vol.13, No.3, pp.317-327, 2000.
- [6] K. Funahashi, “On the approximate realization of continuous mappings by neural networks,” *Neural networks*, Vol.2, No.3, pp.183-192, 1989.
- [7] R. Hecht-Nielsen, *Neurocomputing*, Addison-Wesley Publishing Company, Reading, Massachusetts, 1990.
- [8] A. Hirose, *Complex-valued neural networks*, 2nd ed., Springer, 2012.
- [9] 池口徹, 山田泰司, 小室元政, 合原一幸 (編), *カオス時系列解析の基礎と応用*, 産業図書, 2000.
- [10] M.S. Kim and C.C. Guest, Modification of backpropagation networks for complex-valued signal processing in frequency domain, *Proc. IJCNN*, Vol.3, pp.27-31, 1990.

-
- [11] H. Leung and S. Haykin, The complex backpropagation algorithm, IEEE Trans. Signal Process., Vol.39, no.9, pp.2101-2104, 1991.
- [12] M.A. Little, P.E. McSharry, S.J. Roberts, D.A.E. Costello, and I.M. Moroz, Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection, BioMedical Engineering OnLine 2007, 6:23 (26 June 2007).
- [13] E.N. Lorenz, “Deterministic nonperiodic flow,” Journal of the Atmospheric Sciences, Vol.20, No.2, pp.130-141, 1963.
- [14] D.G. Luenberger, Linear and nonlinear programming, Addison-Wesley Publishing Company, Reading, Massachusetts, 1984.
- [15] 中野良平, ニューラル情報処理の基礎数理, 数理工学社, 2005.
- [16] 新田徹, 古谷立美, “複素バックプロパゲーション学習,” 情報処理学会論文誌, Vol.32, No.10, pp.1319-1329, 1991.
- [17] T. Nitta and M. Tanaka, Current status of research on neural networks with high-dimensional parameters, Circulars of the Electrotechnical Laboratory, 228, 1999.
- [18] T. Nitta, “Local minima in hierarchical structures of complex-valued neural networks,” Neural Networks, Vol.43, pp.1-7, 2013.
- [19] J. Nocedal, and S. Wright, Numerical optimization, Springer Science and Business Media, 2006.
- [20] C. A. Popa, “Quasi-Newton learning methods for complex-valued neural networks,” Proc. IJCNN, Vol.3, pp.1-8, 2015.
- [21] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, CALIFORNIA UNIV SAN DIEGO LA JOLLA INST FOR COGNITIVE SCIENCE, No. ICS-8506, 1985.
- [22] K. Saito, and R. Nakano, “Partial BFGS update and efficient step-length calculation for three-layer neural networks,” Neural Comput., Vol.9, No.1, pp.239-257, 1997.

-
- [23] S. Satoh, and R. Nakano, “Eigen Vector Descent and Line Search for Multilayer Perceptron,” Proc. Int. Multi-Conf. of Engineers and Comput. Scientists, Vol.1, pp1-6, 2012.
- [24] 佐藤聖也, 中野良平, “複素特異階段追跡法の解品質の改善,” 計測自動制御学会 システム・情報部門 学術講演会, pp.820-824, 2014.
- [25] E. K. Chong, and S. H. Zak, An introduction to optimization, Vol.76, John Wiley & Sons, 2013.
- [26] T. Shinbrot, C. Grebogi, J. Wisdom, and J.A. Yorke, “Chaos in a double pendulum,” American Journal of Physics, Vol.60, No.6, pp.491-499, 1992.
- [27] H.J. Sussmann, “Uniqueness of the weights for minimal feedforward nets with a given input-output map,” Neural Networks, Vol.5, No.4, pp.589-593, 1992.
- [28] 鈴木真矢, 中野良平, “複素 BFGS 法を用いた複素ニューラルネットワークの学習法,” 信学論 (D) , Vol.J96-D, No.3, pp.423-431, 2013.
- [29] S. Watanabe, “A formula of equations of states in singular learning machines,” Proc. Int. Joint Conf. on Neural Networks, pp.2099-2106, 2008.
- [30] S. Watanabe, Algebraic geometry and statistical learning theory, Cambridge University Press, 2009.
- [31] S. Watanabe, “A widely applicable Bayesian information criterion,” Journal of Machine Learning Research, Vol.14, pp.867-897, 2013.

研究業績

学術雑誌（査読有り）

- [1] Seiya Satoh, Ryohei Nakano, “Fast and stable learning utilizing singular regions of multilayer perceptron,” Neural Processing Letters, Vol.38, No.2, pp.99-115, 2013.
- [2] 佐藤聖也, 中野良平, “特異領域を利用した多層パーセプトロン探索法の探索枝刈り,” 電子情報通信学会論文誌, Vol.J97-D, No.2, pp.330-340, 2014.

学術雑誌（招待論文）

- [1] Seiya Satoh, Ryohei Nakano, “Multilayer perceptron learning utilizing reducibility mapping,” Studies in Computational Intelligence, Vol.465, pp.261-275, 2012.

国際会議（査読有り）

- [1] Ryohei Nakano, Seiya Satoh, and Takayuki Ohwaki, “Learning method utilizing singular region of multilayer perceptron,” Proc. 3rd International Conf. on Neural Computation Theory and Applications (NCTA), pp.106-111, 2011.
- [2] Seiya Satoh, and Ryohei Nakano, “Eigen vector descent and line search for multilayer perceptron,” Proc. IAENG International Conf. on Artificial Intelligence and Applications (ICAIA), Vol.1, pp.1-6, 2012.
- [3] Seiya Satoh, and Ryohei Nakano, “Multilayer perceptron learning utilizing singular regions and search pruning,” Proc. International Conf. on Machine Learning and Data Analysis (ICMLDA), pp.790-795, 2013.

-
- [4] Seiya Satoh, and Ryohei Nakano, "Complex-valued multilayer perceptron search utilizing singular regions of complex-valued parameter space," Proc. 24th International Conf. on Artificial Neural Networks (ICANN), Vol.8681, pp.315-322, 2014.
 - [5] Seiya Satoh, and Ryohei Nakano, "Singularity stairs following with limited numbers of hidden units," Proc. 6th International Conf. on Neural Computation Theory and Applications (NCTA), pp.180-186, 2014.
 - [6] Seiya Satoh, and Ryohei Nakano, "Complex-valued multilayer perceptron learning using singular regions and search pruning," Proc. International Joint Conf. on Neural Networks (IJCNN), pp.1-6, 2015.
 - [7] Seiya Satoh, and Ryohei Nakano, "A yet faster version of complex-valued multilayer perceptron learning using singular regions and search pruning," Proc. 7th International Conf. on Neural Computation Theory and Applications (NCTA), pp.122-129, 2015.

国内学会（査読無し）

- [1] 佐藤聖也, 大脇高之, 中野良平, "多層パーセプトロンの特異領域を利用した探索法," 信学技報, NC2010-81, pp.85-90, 2010.
- [2] 佐藤聖也, 中野良平, "多層パーセプトロンにおける固有ベクトル降下法と直線探索," 信学技報, NC2011-87, pp.19-24, 2011.
- [3] 佐藤聖也, 中野良平, "特異領域を活用した多層パーセプトロン探索法," 信学技報, NC2012-94, pp.1-6, 2013.
- [4] 佐藤聖也, 中野良平, "複素パラメータ空間の特異領域を利用した複素多層パーセプトロン探索法," 第3回コンピューテーショナル・インテリジェンス研究会, pp.103-108, 2013.
- [5] 佐藤聖也, 中野良平, "出力が実数の問題に特化した目的関数を用いた複素多層パーセプトロン探索," 計測自動制御学会 システム・情報部門 学術講演会, pp.627-632, 2013.

-
- [6] 佐藤聖也, 中野良平, “ 隠れユニット数の範囲を限定した特異階段追跡法, ” 信学技報, NC2013-65, pp.69-74, 2013.
 - [7] 佐藤聖也, 中野良平, “ 探索枝刈りによる複素多層パーセプトロン探索の高速化, ” 第5回コンピューターショナル・インテリジェンス研究会, pp.91-96, 2014.
 - [8] 佐藤聖也, 中野良平, “ 複素特異階段追跡法の解品質の改善, ” 計測自動制御学会システム・情報部門 学術講演会, pp.820-824, 2014.
 - [9] 佐藤聖也, 中野良平, “ 特異階段追跡法の一層の高速化と解品質の改善, ” 信学技報, NC2014-119, pp.289-294, 2015.
 - [10] 佐藤聖也, 中野良平, “ 複素多層パーセプトロン学習における一層の探索枝刈りの効果, ” 第7回コンピューターショナル・インテリジェンス研究会, pp.37-42, 2015.
 - [11] 佐藤聖也, 中野良平, “ 複素多層パーセプトロンを用いたカオスの時系列の予測, ” 第8回コンピューターショナル・インテリジェンス研究会, pp.1-6, 2015.

学術表彰

- [1] Seiya Satoh, and Ryohei Nakano, Best student paper award of the 2012 IAENG International Conf. on Artificial Intelligence and Applications, 2012.
- [2] 佐藤聖也, 電子情報通信学会東海支部 学生研究奨励賞 (修士), 2013.
- [3] Seiya Satoh, IEEE Computational Intelligence Society Japan Chapter Young Researcher Award, 2014.

その他

- [1] 独立行政法人 日本学術振興会 特別研究員 DC2, 2014.4 ~ 2016.3.

実数と複素数の多層パーセプトロンの特異領域を利用した探索法の研究

TP13801 佐藤 聖也

(中部大学工学研究科情報工学専攻)

2016 年 2 月
