CMA-ES を用いたリカレントニューラルネットの構造と重みの同時最適化に関する実験的考察

○石川卓実　永田裕一 小野典彦 （徳島大学）

On Optimization of Recurrent Neural Network Topologies Along with Weights Using CMA-ES

∗T. Ishikawa Y. Nagata and N. Ono (Tokushima University)

Abstract– Recently, various methods called TWEANN have been proposed that can simultaneously optimize the topology and weight of a recurrent neural network. However, most of them require a large number of parameter settings when applied and lack the ability to optimize weights. In this paper, we propose a new TWEANN and its variant that can optimize weights using a powerful function optimization method such as CMA-ES, without requiring numerous parameter settings, and show their potentials and limitations through application to a couple of non-Markovian tasks.

Key Words: TWEANN, リカレントニューラルネット, CMA-ES

1　はじめに

リカレントニューラルネット (RNN) は，制御系やソフトウェアエージェントの設計および時系列データ予測などの幅広い分野に応用可能な有用な枠組みである．しかしながら，高性能の RNN を設計するためには，その構造および結合の重みを同時に決定する必要があり，これを人手により行うことは困難である．近年，RNN の構造と結合の重みを同時にかつ進化的に設計可能な TWEANN(Topology and Weight Evolution Artificial Neural Networks)[1] と呼ばれる種々の手法 (NEAT[1]，MBEANN[2]，CMA-TWEANN[3]，ANS(μ, λ)-TWEANN[4][5] など) が提案され，注目を集めている．

しかし，それらはいずれも RNN の構造を変異させるための遺伝的オペレータと重みを変異させるための数値的オペレータを組合せた複雑な手法となっており，応用時に多数のパラメータ設定が必要なものがほとんどである．また，それらの多くは重みの最適化能力に欠け，その性能には限界がある．森口らの CMA-TWEANN は，RNN の構造の探索空間を急速に絞り込むと同時に，その重みを CMA-ES[6] で最適化することにより，他の TWEANN に比べ，重みの最適化能力に長け圧倒的に高速であるが，その構造の最適化能力には疑問がある．

本研究では，(1) より単純で応用の際に多数のパラメータ設定を必要としない TWEANN および (2) それを発展させ，RNN の構造の探索空間を緩やかに絞り込みながら，その構造と重みを同時に最適化可能な TWEANN を提案する．また，ゲームのソフトウェアエージェント設計問題である Mario AI Benchmark[7] およびマルチエージェント系設計問題である連続型追跡問題への適用を通して，それらの性能に関する実験的考察を行う．

2　構造と重みのシームレスな設計手法

ここでは RNN の “構造と重みのシームレスな設計手法” と呼ぶ TWEANN を提案する．従来手法は，RNN の構造を変異させるための遺伝的オペレータと重みを変異させるための数値的オペレータを組み合わせた複雑な手法となっており，応用時に多数のパラメータ設定が必要である．また，それらの多くは最新の最適化手法を取り入れていないため最適化能力に欠ける．本提案手法は単純でありながら，CMA-ES のような強力な関数最適化手法を用いて， RNN の構造と重みをシームレスに最適化可能で，多数のパラメータ設定を必要としないものとなっている．

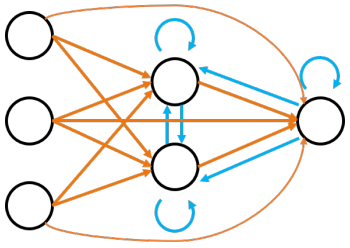

Fig. 1: 最大構造 RNN

2.1　アルゴリズム

本手法は，解となる RNN の最大構造をあらかじめ設定しておき，RNN を構成する各結合の有無および重みを同時に最適化することで良好な解を特定する手法である．

1. 最大構造 RNN の設定

   RNN の層の数と各層のノードの上限数を設定し，多層フィードフォワード型ニューラルネットの中間層と出力層にセルフループを含む任意のリカレント結合を付与した RNN を生成し，それを “最大構造 RNN” とする．また，この RNN が含む結合の数を M とする．最大構造 RNN の例を Fig. 1 に示す．

2. 重みベクトルの生成

   最大構造 RNN が含むすべての結合にラベル $(1, 2, 3, ..., M)$ を付与し，そのラベルに対応する結合の重みを格納した実数値の重みベクトル $\mathbf{w}$ を生成する．

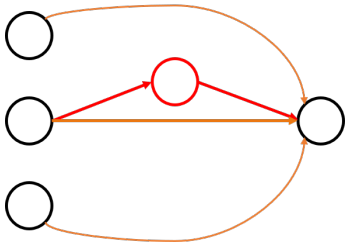

Fig. 2: AddNode オペレータを適用した RNN

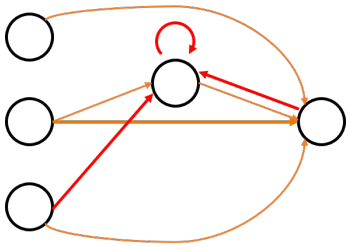

Fig. 3: AddEdge オペレータを適用した RNN

$$\mathbf{w} = [w_1, w_2, w_3, ..., w_M] \tag{1}$$

3. 構造ベクトルの生成

   ラベルに対応した結合の有無を決定するための実数値の構造ベクトル $\mathbf{t}$ を生成する．ここで，$t_i \geq \varepsilon$ のとき，結合 i が存在するものとする．

$$\mathbf{t} = [t_1, t_2, t_3, ..., t_M] \tag{2}$$

4. 最適化

   重みベクトルの末尾に構造ベクトルを連結した解ベクトル $\mathbf{x}$ を生成する．進化戦略 CMA-ES を用いて，終了条件を満たすまで解ベクトル $\mathbf{x}$ を最適化する．

$$\mathbf{x} = [w_1, w_2, w_3, ..., w_M, t_1, t_2, t_3, ..., t_M] \tag{3}$$

3 緩やかな構造変異による設計手法

ここでは “緩やかな構造変異による設計手法” と呼ぶ TWEANN を提案する．構造と重みのシームレスな設計手法では，解ベクトルが高次元となるため，その最適化が困難になってしまう．本提案手法では，この問題点を回避するため，(i) 最大構造 RNN を小さなもの（たとえば，2 層パーセプトロン）に初期化しておき，(ii) 構造と重みのシームレスな設計手法を用いて RNN を最適化し，(iii) 良好な解が特定できなければ最大構造 RNN を少しだけ拡大して，さらに (ii) を繰り返すという方法をとる．また，最大構造 RNN の拡大は，CMA-TWEANN で採用されている AddNode および AddEdge と同様のオペレータを確率的に適用することで実現する．なお，ステップ (ii) で存在を否定された結合は，それ以降，最大構造 RNN には付与しないものとする．

3.1 AddNode

Fig. 2 に示すように，現在の最大構造 RNN が含むノードの間に確率 $P(\alpha_n)$ で新たなユニットおよび 2 つの結合を付与する．

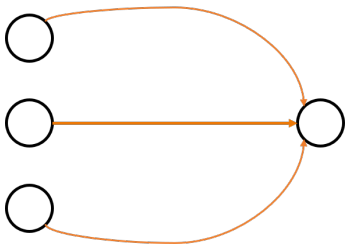

Fig. 4: 最小構造の RNN

3.2 AddEdge

Fig. 3 に示すように，現在の最大構造 RNN が含むノードの間に新たな結合を確率 $P(\alpha_e)$ で付与する．

3.3 アルゴリズム

1. 初期化

   最大構造 RNN を 2 層パーセプトロンに初期化する．

2. 最適化

   構造と重みのシームレスな設計手法を用いて RNN を最適化し，終了条件を満たしていれば終了する．なお，CMA-ES の最適化による世代数 g は $g = d \cdot \alpha_g$ (d は次元数，α_g は実数値のパラメータ) とする．

3. 構造の拡大

   ステップ 2 の最適化終了後に評価値が最良の個体の構造に，AddNode および AddEdge を適用して得られる構造を新たな最大構造 RNN として，ステップ 2 に戻る．

4 Mario AI Benchmark へ適用

4.1 実験方法

Mario AI Benchmark はゲームのソフトウェアエージェント設計問題である．マップの長さは 256 セル，高さは 15 セル，制限時間は 200 タイムとなっており，ステージには難易度，シード値を設定できる．

ここでは，マップの一部の情報しか観測できない RNN で Mario の行動政策を表現するものとして，その構造と重みの最適化を試みる．解となる RNN の層の数の上限 3，各層のノードの数は入力層 58 (敵情報 25, マップ情報 25, ジャンプ判定, ファイア判定, 接地判定, 穴判定 5), 出力層 5 (右, 左, 下, ジャンプ, ダッシュの 5 個のキーの組み合わせ 9 通り)，中間層の上限 10 とする．

個体の評価はマリオがスタート時点からゴール地点まで進んだ距離により決定し，これを最大化することを目的とする．マリオがゴール地点に到達したとき成功とする．一方，制限時間内にゴール地点に到達できなかった，敵に接触した，穴に落ちたとき失敗したとみなす．

学習は，各世代ごとにシード値をランダムに設定した環境 15 個を各個体に与え，各世代の終了時にその世代で最も評価値の良い個体を用いて，汎化テスト用の全世代共通の環境 30 個で実行して性能を計測する．

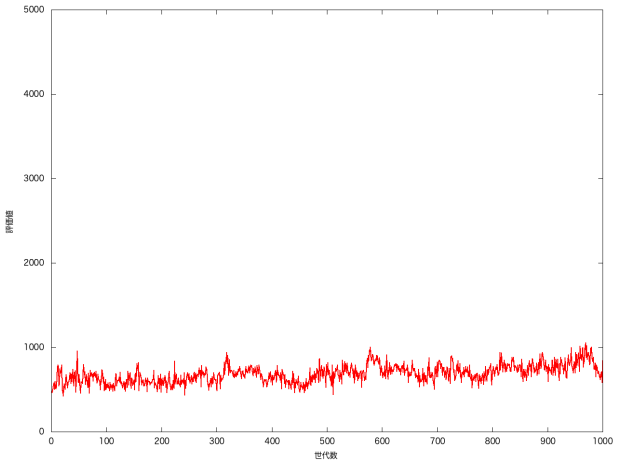

Fig. 5: NEAT の評価値（Mario AI Benchmark：難易度 0）

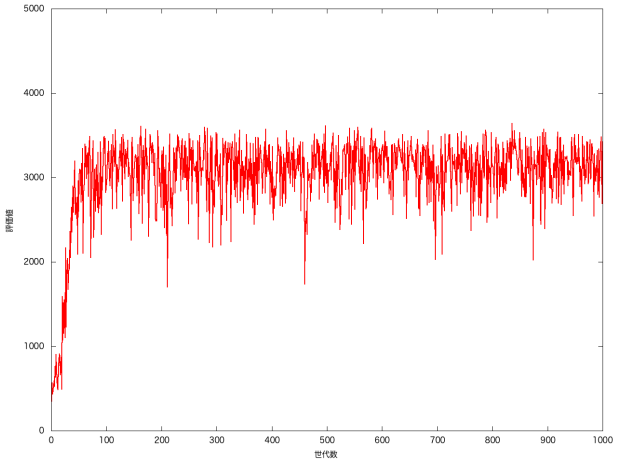

Fig. 6: 重みと構造のシームレスな設計手法の評価値（Mario AI Benchmark：難易度 0）

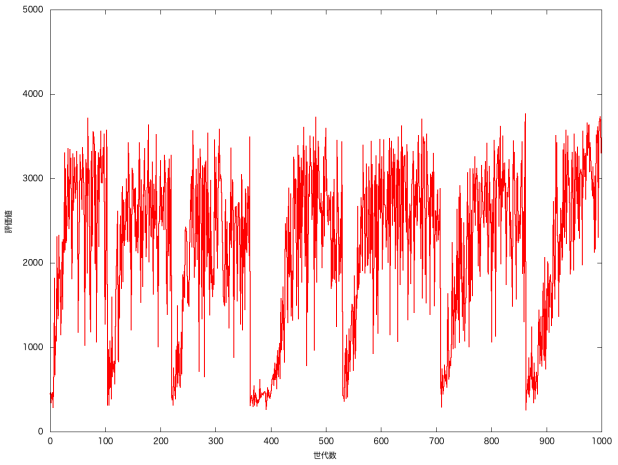

Fig. 7: 緩やかな構造変異による設計手法の評価値（Mario AI Benchmark：難易度 0）

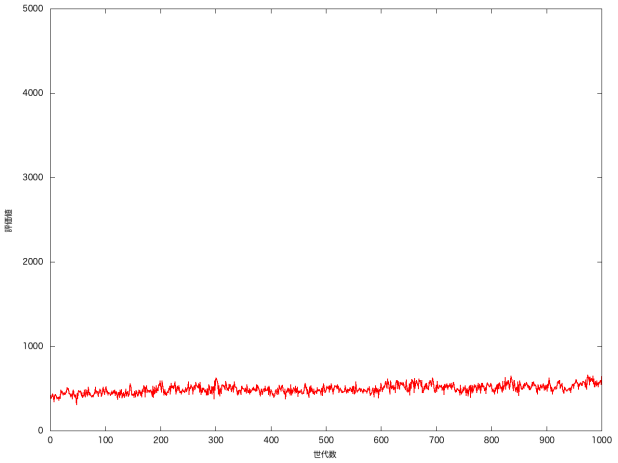

Fig. 8: NEAT の評価値（Mario AI Benchmark：難易度 1）

4.2 難易度 0 の実験結果

難易度 0 における実験結果を Fig. 5–7 に示す．両提案手法ともに代表的な TWEANN である NEAT よりも良好な評価値の個体を特定できた．

4.3 難易度 1 の実験結果

難易度 1 における実験結果を Fig. 8–10 に示す．難易度 0 より問題が難しくなっているため，いずれの手法においても評価値は低下しているが，両提案手法ともに NEAT よりも良好な評価値の個体を特定できた．

5 連続型追跡問題へ適用

5.1 実験方法

追跡問題は，複数のハンターが 1 匹の獲物を追跡･捕獲するマルチエージェント系設計のためのベンチマーク問題である．ここでは 4 匹のハンターを用いて，マップサイズが 200×200 の連続型追跡問題を扱う．ハンターの行動は共通の RNN により決定する．Mario AI Benchmark の場合と同様，RNN はマップの一部しか観測できないため，非マルコフ的な環境上で行動することとなる．

ここでは，各ハンターに座標，方位，視野を設定し，ニューラルネットの入力は視野内に存在するハンターあるいは獲物との相対視野である．視野は各ハンターを中心としてマップの一定の範囲（本実験ではマップの 40%）の情報が得られるものとする．出力は 2 種類の設定があり，1 つ目は，ハンターの向いている方角へ進む距離 $\in \{0, 1\}$，回転する角度 $\in \{-\pi/4, 0, \pi/4\}$ の離散値出力により行動を決定する．2 つ目はハンターの向いている方角へ進む距離 $\in [0, 1]$，回転する角度 $\in [-\pi/4, \pi/4]$ の実数値出力により行動を決定する．

ニューラルネットの層数は上限 3，各層のノード数は入力層 8 (獲物と他のハンターの相対座標), 出力層 5（離散値出力のとき） または 2（実数値出力のとき），中間層は上限 5 とする．

個体の評価値は捕獲までに有したステップ数により決定し，これを小さくすることを目的とする．捕獲条件はターゲットの一定の範囲内にすべてのハンターが存在しているときに成功したとする．一方，10,000 ステップ経過しても捕獲条件を満たすことができなかった場合，捕獲は失敗したとみなす．

学習は，各世代ごとにハンターおよびターゲットの初期座標をランダムに配置した環境 10 個を各個体に与え，各世代の終了時にその世代で最も評価値の良い個体を用いて，汎化テスト用の全世代共通の環境 20 個で実行して性能を計測する．

5.2 離散値出力の実験結果

離散値出力における実験結果を Fig. 11–13 に示す．両提案手法ともに NEAT よりも良好な評価値の個体を

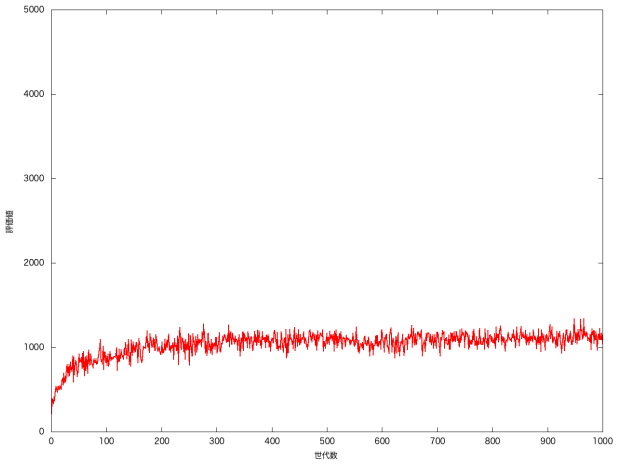

Fig. 9: 重みと構造のシームレスな設計手法の評価値（Mario AI Benchmark：難易度 1）

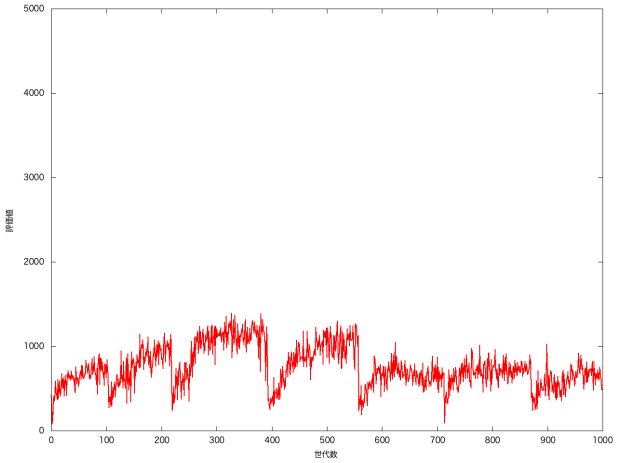

Fig. 10: 緩やかな構造変異による設計手法の評価値（Mario AI Benchmark：難易度 1）

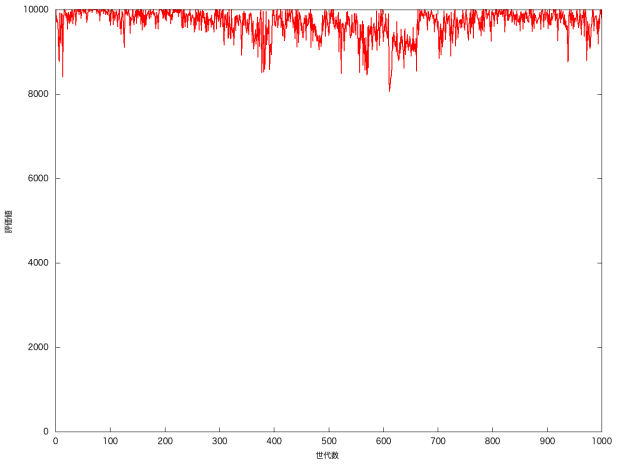

Fig. 11: NEAT の評価値（連続型追跡問題：離散値出力）

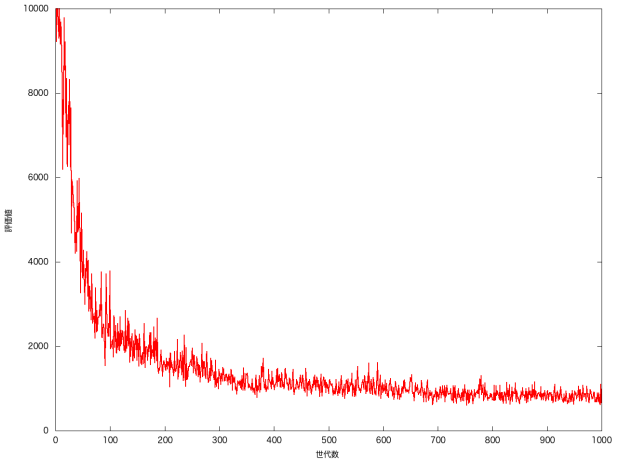

Fig. 12: 重みと構造のシームレスな設計手法の評価値（連続型追跡問題：離散値出力）

特定できた．また，両提案手法ともに最良評価値に大きな差はないが，Fig. 12 では 1000 世代付近で最良評価値をマークしており，Fig. 13 では 250 世代付近でも最良評価値をマークしている．このことから，緩やかな構造変異による設計手法の特徴である次元数の削減が確認でき，学習の高速化の可能性を示すことができた．

5.3 実数値出力の実験結果

実数値出力における実験結果を Fig. 14–16 に示す．NEAT と緩やかな構造変異による設計手法では評価値がわずかに上昇している世代はあるが，良好といえる個体は獲得できていない．構造と重みのシームレスな設計手法は，評価値が学習の初期の段階で収束している．しかし，離散値出力の実験においては評価値の改善がされ続け，最終的には評価値 400 程度の解を特定できているのに対し，こちらの実験では評価値が 4000 程度に停滞した．これは，出力が離散値から実数値へと変更されたことにより，行動の最適化が難しくなったためと考えられる．

6 考察

Mario AI Benchmark および連続型追跡問題において，提案手法である構造と重みのシームレスな設計手法は，代表的な TWEANN である NEAT よりも良好な解を得ることができた．また，緩やかな構造変異による設計手法も連続型追跡問題の実数値出力版での実験を除き NEAT よりも良好な解を得ることができた．実数値出力版の連続型追跡問題では，構造と重みのシームレスな設計手法でのみ評価値の改善がみられたが，緩やかな構造変異による設計手法では， NEAT と同様，良好な性能を示すことはできず，評価値の改善もほとんどみられなかった．この実験における構造と重みのシームレスな設計手法と緩やかな構造変異による設計手法の構造サイズの推移を Fig. 17 および Fig 18 に示す．Fig. 17 では，構造サイズがどの世代も 60 程度に推移しているのに対し，Fig. 18 では最も大きい構造サイズをマークしている世代でも 40 程度の大きさである．このことから，問題を解くのに十分な大きさの構造サイズを有していなかったため，評価値が改善しなかったと考えられる．よって，評価値の改善が長期に渡って見られない場合，構造サイズを大きく拡大するような機能を取り入れることによって性能を改善できる可能性がある．

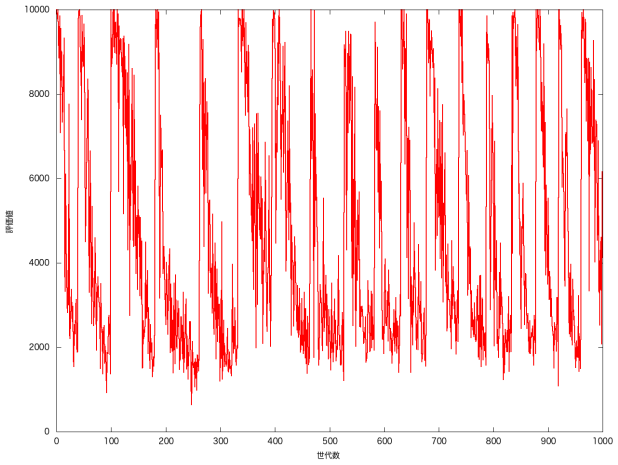

Fig. 13: 緩やかな構造変異による設計手法の評価値（連続型追跡問題：離散値出力）

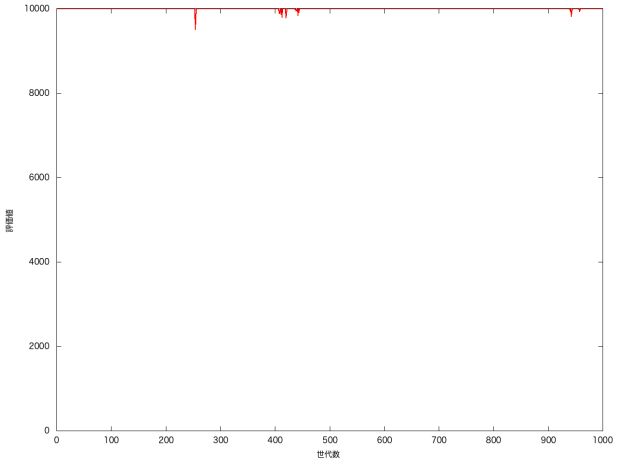

Fig. 14: NEAT の評価値（連続型追跡問題：実数値出力）

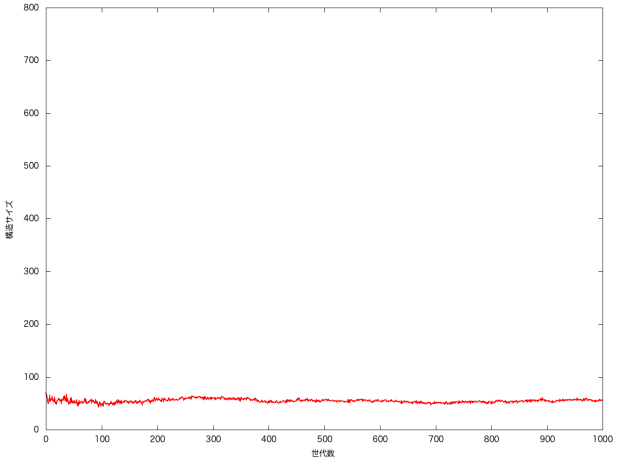

Fig. 17: 構造と重みのシームレスな設計手法の構造サイズ（連続型追跡問題：実数値出力）

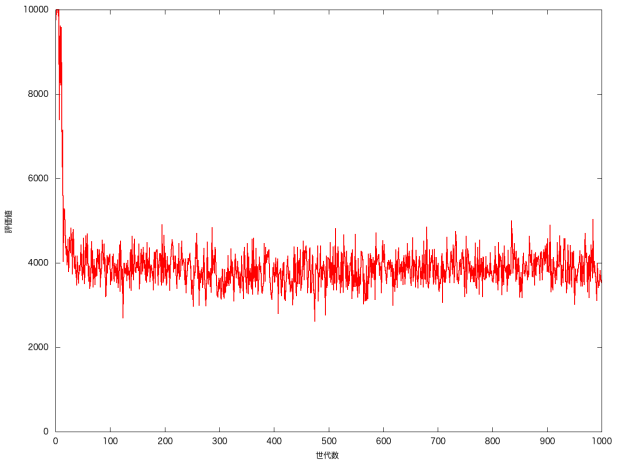

Fig. 15: 重みと構造のシームレスな設計手法の評価値（連続型追跡問題：実数値出力）

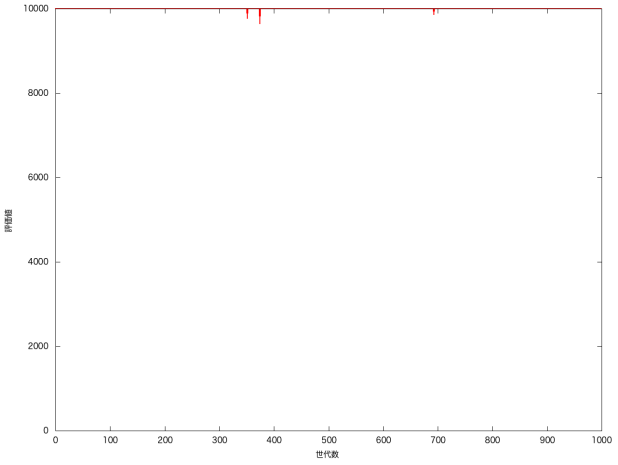

Fig. 16: 緩やかな構造変異による設計手法の評価値（連続型追跡問題：実数値出力）

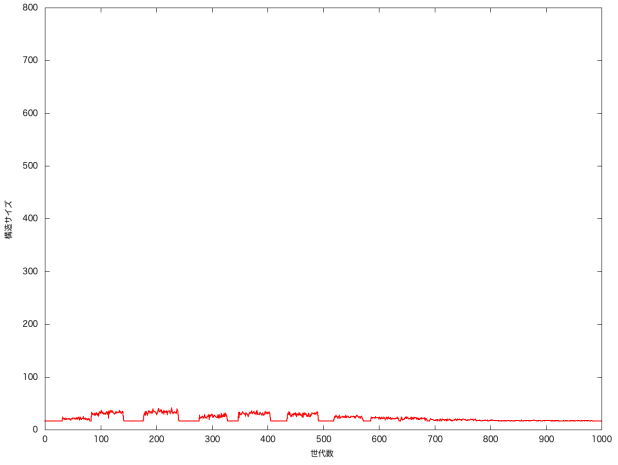

Fig. 18: 緩やかな構造変異による設計手法の構造サイズ（連続型追跡問題：実数値出力）

7　おわりに

近年，リカレントニューラルネット（RNN）の構造と重みを同時に最適化可能な TWEANN と呼ばれる種々の手法が提案されている．しかし，それらはいずれも RNN の構造を変異させるための遺伝的オペレータと重みを変異させるための数値的オペレータを組み合わせた複雑な手法となっており，応用時に多数のパラメータ設定が必要となる．また，それらの多くは重みの最適化能力に欠け，性能に限界がある．本研究では，より単純で多数のパラメータ設定を必要とせず，CMA-ES などの強力な関数最適化手法を用いて重みの最適化が可能な新たな TWEANN を提案するとともに，Mario AI Benchmark および連続型追跡問題への適用を通じて，その有効性と課題に関する実験的考察を行った．

参考文献

1) K. Stanley and R. Miikkulainen: Evolving neural networks through augmenting topologies; Evolutionary Computation, Vol. 10, No. 2, pp. 99–127 (2002)
2) 大倉 和博，岩波 悟史 : 進化型人口神経回路網の構造進化のための一手法 -二重倒立振子問題への適用，システム制御情報学会論文誌，Vol.19，No.2，pp.51–58，2006.
3) 森口 博貴，本位田 真一 : CMA-TWEANN: 効率的なトポロジ進化型ニューラルネットの提案および性能評価，第 2 回進化計算学会研究会/第 8 回進化計算フロンティア研究会 合同研究会資料，pp.179–188，2012.
4) 堀田 駿仁，西村 悠哉，小野 典彦，永田 裕一 : 構造と重みのシームレスな変異と解の多様性維持に基づくリカレントニューラルネットの進化的設計，計測自動制御学会第 42 回知能システムシンポジウム資料，2015.
5) 西村 悠哉，小野 典彦，永田 裕一 : 構造のシームレスな変異と解の多様性維持に基づくリカレントニューラルネットの進化的設計，進化計算学会，進化計算シンポジウム 2016 講演論文集，2016.
6) N. Hansen and A. Ostermeier: Completely derandomized self-adaptation in evolution strategies; Evolutionary Computation, Vol. 9, No. 2, pp. 159–195 (2001).
7) 2012 Mario AI Championship
http://www.marioai.org/