自己組織化適応制御器（SOAC）の学習と制御性能に関する検討：多重順逆対モデル（MPFIM）との比較

An Investigation Regarding Learning and Control Performances Between the SOAC and the MPFIM

湊原 哲也 (PY)†, 古川 徹生 ‡

Tetsuya Minatohara(PY), and Tetsuo Furukawa

† 津山工業高等専門学校

‡ 九州工業大学大学院生命体工学研究科

minato@tsuyama-ct.ac.jp

Abstract— This study aims at a comparison of a control performance between the multiple paired forward and inverse models (MPFIM) and the self-organizing adaptive controller (SOAC) proposed by the authors. Study results show that the performance of the SOAC is better than that of the MPFIM in all cases we examined.

Keywords— **modular network SOM, self-organizing adaptive controller, multiple paired forward and inverse models, feedback-error-learning, motor control**

1 はじめに

自己組織化適応制御（Self-Organizing Adaptive Controller : SOAC）はモジュラーネットワーク型自己組織化マップ（modular network SOM : mnSOM）[?] の機能モジュールとして順モデル（予測器）と逆モデル（制御器）の対を採用したものである [?]．したがって SOAC は mnSOM を運動制御へ応用したものと見なすことができ，複数の環境や制御対象が存在する課題に対して有効に働く．このような運動制御を目的とした他の手法としては mixture of experts[?] に Kawato らのフィードバック誤差学習を導入したモデル [?] や Wolpert and Kawato の多重順逆対モデル（Multiple Paired Forward and Inverse Models : MPFIM）[?] がある．

本研究では SOAC の学習と制御性能について類似手法である MPFIM との比較によって検討を試みる．

2 多重順逆対モデルと自己組織化適応制御器

一般にモジュラーネットワークでは複数存在するモジュールのうちどのモジュールを用いるか（モジュール選択），およびモジュールをどのように学習するか（モジュール学習）の 2 つが問題となる．多重順逆対モデル（以下 MPFIM）は順モデル（予測器）と逆モデル（制御器）の対をモジュールとしており，順モデルの予測誤差をモジュールの選択と学習に用いている．すなわち予測誤差の soft-max によって “責任信号” を計算し，各モジュールの出力に重みづける．したがって責任信号の大きいモジュールほど学習と制御に大きく貢献する．

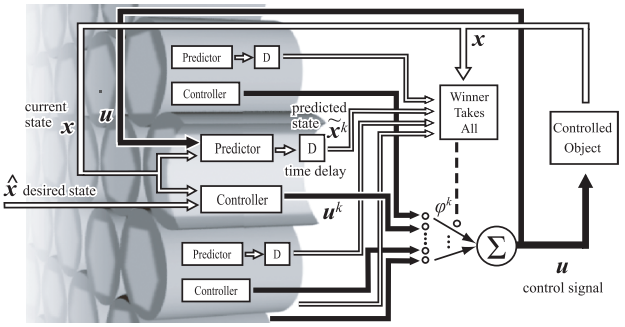

図 1: 自己組織化適応制御器（SOAC）の構成

一方，我々の自己組織化適応制御器（以下 SOAC）は MPFIM と同様に予測器と制御器を対とするモジュール構造（図??）を持つが，責任信号の決め方が MPFIM とは決定的に異なる．SOAC では Kohonen の自己組織化マップ（Self-Organizing Map : SOM）の近傍関数に基づいて責任信号を算出する．すなわち予測誤差を最小としたモジュール（Best Matching Module : BMM）が責任信号が最大となり，マップ空間において BMM からの距離が遠いモジュールほど責任信号の値は小さくなる．この近傍関数の導入により SOAC は単に各モジュールを学習できるだけでなく，モジュール間の類似関係を読み取れる “システムの地図” を生成できる．言うまでもなく MPFIM にそのような機能は備わっていない．

以下に SOAC のアルゴリズムについて概略を述べる．詳細については文献 [?] を参照されたい．SOAC のアルゴリズムは各予測器を学習するための学習モードと実際に制御を行うための実行モード（ただし制御器の学習も同時に行う）の 2 つから構成される．したがって予測器を学習するための学習データが事前に必要となる．

学習モードではまず各予測器の出力と学習データ間に生じる予測誤差を各クラス（制御対象）ごとに評価する．次に予測誤差を最小としたモジュールを BMM とする．すべてのクラスに対する BMM が決まったら近

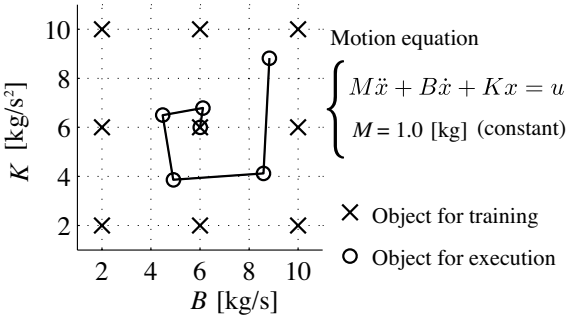

図 2: 制御対象のパラメータ

傍関数によって学習分配率を求める．この学習分配率がMPFIM における責任信号に相当する．最後に通常の誤差逆伝搬学習に学習分配率を重みづけることで各予測器の学習を行う．これらの行程を近傍関数の径を小さくしながらネットワークが定常状態になるまで繰り返す．

予測器の学習が終わったら実行モードへと移る．実行モードでは制御器の学習と対象の制御を同時に行う．実行モードにおいても各モジュールの出力と学習は責任信号によって重みづけられる．最終的に各予測器が表現する順モデルに対する逆モデルが各制御器によって実現される．その結果，SOAC は環境（制御対象）の変化に合わせ BMM を切り替え，様々な環境に適応可能となる．

3　シミュレーション

本研究では学習により獲得されたモジュールを用いて MPFIM と SOAC の制御性能の比較を行う．シミュレーションで用いた制御対象は図**??**の通りである．

3.1　シミュレーションの条件と結果

物理特性の異なる 9 種類の制御対象を用いて MPFIM と SOAC の学習をそれぞれ行い，6 種類の制御対象を用いて性能を評価する．ただし，評価に用いる制御対象はただ 1 つを除いて学習では用いていない未知の制御対象とする（図**??**）．制御性能の評価として目標軌道と実軌道との間に生じる二乗平均平方根誤差（Root Mean Square Error : RMSE）を用いる．上記の制御対象に対し，ネットワークの初期値を毎回変えて学習・制御を 100 回試行をおこなうときの RMSE の平均と標準偏差を制御対象の数に対しモジュールの数が少ない（1, 4），同じ（9），多い（16，25）の 3 つの場合について調べる．

結果を図**??**に示す．MPFIM を用いた場合，制御性能が最も高いのはモジュール数 K が制御対象の数 I に等しいとき（K=I=9）であった．一方 SOAC を用いた場合，制御性能はモジュール数の増加に伴い単調に向上した．したがってモジュール数 $K = 25$ のときに制御性能が最も高かった．また両手法の制御性能を同じモジュール数の元で比較したところ，いずれの場合においても制御性能が高いのは SOAC であった．さらに，MPFIM

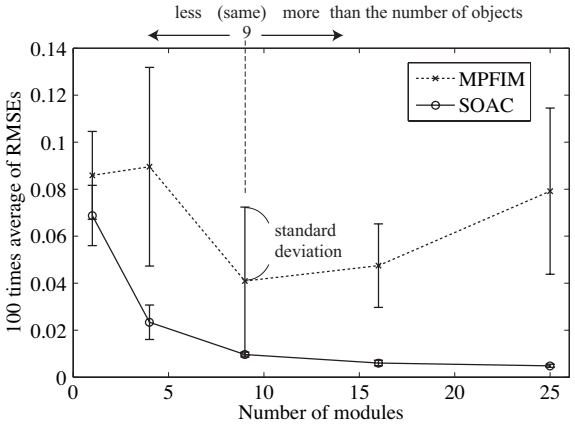

図 3: シミュレーション結果．初期条件を変え学習・制御を 100 回試行したときの MPFIM（破線）と SOAC（実線）の二乗平均平方根誤差（RMSE）の平均と標準偏差を表す．

は制御性能に関する試行ごとのばらつきが大きく，学習結果が初期値に大きく依存したのに対し，SOAC は試行ごとのばらつきが小さく，学習結果が安定して得られていることが図**??**より分かる．

4　おわりに

本研究では SOAC と MPFIM の性能比較として，制御対象の数がモジュール数よりも少ない場合と多い場合で両手法の制御性能がどのように変化するかついて調べた．その結果，SOAC は MPFIM に対しすべてのケースにおいて制御性能が高かった．今後は，オンライン学習へ向けた SOAC の学習・制御アルゴリズムの修正と SOAC の実機への応用を考えている．

参考文献

[1] 徳永憲洋，肝付謙二，安井湘三，古川徹生 (2005) “関数空間型 SOM.” 日本神経回路学会誌, **12** , **1** , 39–51.

[2] 湊原哲也，古川徹生 (2008) “適応性と汎化性を考慮した自己組織化適応制御器.” 電子情報通信学会論文誌, **J91-D**, **4**, 1142–1149.

[3] R.A. Jacobs and M.I. Jordan (1991) “Adaptive mixtures of local experts.” Neural Computation, **3**, 79–87.

[4] H. Gomi and M. Kawato (1993) “Recognition of manipulated objects by motor learning with modular architecture networks” Neural Networks, **6**, 485–497.

[5] D.M. Wolpert and M. Kawato (1998) “Multiple paired forward and inverse models for motor control.” Neural Networks, **11**, 1317–1329.