新しい自己組織化マップの創出：
モジュラーネットワーク SOM

2006 年

徳永 憲洋

要旨

Kohonen の自己組織化マップ（Self-Organizing Map: SOM）は高次元空間のベクトルデータを 1, 2 次元程度の低次元空間へトポロジーを保存しつつ写像することにより，ベクトルデータ間の関係を視覚的に捉えられる地図（マップ）として表現できる．SOM はベクトル空間で扱われる手法なのでアーキテクチャ，アルゴリズムは非常にシンプルである．そのため，SOM は提案されて以来，ロボティクス，医療，生物学，音声・画像認識などのさまざまな分野で利用されており，その中で拡張された SOM もいくつか提案されている．このように，これまで多くの応用とアーキテクチャ・アルゴリズムの拡張がなされているにもかかわらず，ほとんどの場合，SOM のユニットはベクトルデータしか扱えず，ベクトル空間においてマップが生成される従来の SOM の原理は残ったままである．

本論文の目的は従来の SOM におけるベクトルユニットをニューラルネットワークなどのモジュールに置き換えたモジュラーネットワーク SOM (Modular Network SOM : mnSOM) を提案するものである．mnSOM はモジュラーネットワークの学習に SOM の基本原理である競合・協調作用を導入したものと考えることができ，各モジュールはコードベクトルだけではなくさまざまな '機能' を実現することができる．すなわち mnSOM は，ただ単に SOM を拡張したものではなく，さまざまなデータタイプに適用できるような一般化された SOM を実現する．もし mnSOM が SOM としての特徴を持ち合わせつつ，時系列データ集合，システム（関数）集合などのさまざまなデータタイプに適用できるならば，工学的にも非常に効果的なツールとなることが期待できる．

この目的に対し，私は以下のことを行った．(1) mnSOM の提案，(2) mnSOM と SOM との等価性の検証，(3) さまざまな課題への応用，の 3 点である．(2) において，SOM アルゴリズムの骨格をそのまま mnSOM で用いても，SOM と mnSOM のマップの性質が等価であることが確認できた．したがって，先行研究における SOM に対する多くの理論的な見解が mnSOM に対しても当てはめることができる．これは mnSOM のアルゴリズムが SOM の枠組みから外れることなくシンプルであるとともに mnSOM における信頼性の向上につながる．さらに，先行研究で提案されているさまざまな SOM アーキテクチャの拡張を mnSOM にも適用できる．これによって，mnSOM はさらに汎用性が高まるだろう．

以上をふまえて本論文では具体的に下記のことを述べる．(1) mnSOM の提案と mnSOM のアーキテクチャ，アルゴリズムについて述べる．(2) 正規直交展開された関数集合を用い，シミュレーションによって SOM および mnSOM で作られる両方のマップの性質が等価であることを述べる．すなわち，SOM と mnSOM の作るマップの性質は等価である証明を示す．(3) さまざまなデータタイプ（気象時系列データ集合，多様体集合，ダイナミカルシステム集合）に対する mnSOM の適用について述べる．まず，気象時系列データ集合に対して mnSOM が，気象時系列データだけから地理的トポロジーを保存し

たマップを生成することを示す．次に，周期振動波形や 3D 物体をさまざまな角度から 2 次元平面に射影した射影像集合のようなベクトル空間上において多様体を形成するデータに対する mnSOM のマップ結果を示す．さらに多様体集合を扱うような課題（パターン認識の分野では存在する）に対して mnSOM が適用できるかを検証する．最後にダイナミカルシステム集合に対して mnSOM が生成するマップを示し，mnSOM が非線形制御やロボティクスなどの課題にも適用できることを示す．

目次

要旨 i

第 1 章 序論 1

1.1 はじめに 1

1.2 研究の目的・意義 2

1.3 本論文の構成 3

第 2 章 基礎知識 5

2.1 Self-Organizing Map 5

2.2 モジュラーネットワーク 14

第 3 章 mnSOM のアーキテクチャとアルゴリズム 21

3.1 はじめに 21

3.2 アーキテクチャおよび本研究における問題の枠組み 22

3.3 アルゴリズム 24

第 4 章 多項式関数集合を用いた mnSOM の有効性の検討 27

4.1 シミュレーションの枠組み 27

4.2 シミュレーションの結果 28

4.3 考察 29

第 5 章 mnSOM による気象ダイナミクスの自己組織化マップ 37

5.1 はじめに 37

5.2 シミュレーションの枠組み 38

5.3 シミュレーション結果 38

5.4 考察 38

第 6 章 mnSOM による NL-ASSOM の実現 45

6.1 はじめに 45

6.2 理論的な枠組み 46
6.3 シミュレーション 49
6.4 考察 55

第 7 章 mnSOM による動的システム集合の自己組織化マップ 57
7.1 動的システム集合を扱うための mnSOM のアーキテクチャ，アルゴリズム 57
7.2 計算機シミュレーション 60
7.3 考察 62

第 8 章 考察 65
8.1 通常型 SOM との比較 65
8.2 他のモジュラーネットワークとの関連 66
8.3 mnSOM が扱う課題の問題設定について 66
8.4 確率密度関数のクラス依存性 67
8.5 ローカルミニマムの問題 68
8.6 一般化された mnSOM について 69

第 9 章 まとめ 73

研究業績リスト 74

参考文献 75

謝辞 79

略語

ANN	Autoassociative Neural Network
3L-ANN	3 Layer Autoassociative Neural Network
5L-ANN	5 Layer Autoassociative Neural Network
ASSOM	Adaptive Subspace SOM
BMM	Best Matching Module
BMU	Best Matching Unit
BP	Back Propargation
LVQ	Learning Vector Quantization
NL-ASSOM	Nonlinear ASSOM
NL-PCA	Nonlinear Principal Component Analysis
MLP	Multilayer Perceptron
mnSOM	Modular Network SOM
PCA	Principal Component Analysis
RBF	Radial Basis Function
RNN	Recurrent Neural Betwork
SOM	Self-Organizing Map
VQ	Vector Quantization
WTA	Winner-Take-All

表記

変数のインデックスは，下付きをデータに関するもの，上付きをユニットあるいはモジュールに関するものにする．

2章

データ	
$\mathcal{A}$	マップ空間
I	データベクトルの数
$\boldsymbol{x}$	データベクトル
SOM	
K	ユニット，モジュールの数
$\boldsymbol{\xi}$	ユニットのマップ空間における座標
$\boldsymbol{m}$	ユニットの実現するコードベクトル
E	評価誤差
k^*	勝者ユニットの番号
α	VQ における修正量
$\phi(\cdot)$	近傍関数
t	学習ステップ数
$\sigma(t)$	t ステップ目の近傍関数の範囲
ψ	規格化されたコードベクトルの修正率
H	1 近傍隣接ユニットの集合
競合モジュラーネットワーク	
K	モジュールの数
$\boldsymbol{y}$	モジュールの出力
$\hat{\boldsymbol{y}}$	モジュラーネットワーク全体の出力
p	モジュールが選択される確率
T	アニーリング温度
$\bar{E}$	誤差の期待値
$\boldsymbol{w}$	モジュールの結合荷重
ϕ	モジュールの学習率

3章以下

データ	
d	特徴空間の次元数
$\mathcal{X}$	特徴空間
$\mathcal{A}$	マップ空間
$\mathcal{D}$	データ集合族（データ集合の集合）
I	クラス数（データ集合の数）
$f_i(\cdot)$	i-th クラスの関数（モデル）
J	データベクトルの数
D	データ集合
C	クラス
$(\boldsymbol{x}, \boldsymbol{y})$	データベクトルのペア（$\boldsymbol{x}$:入力，$\boldsymbol{y}$:出力）
mnSOM	
K	モジュールの数
$\boldsymbol{\xi}$	モジュールのマップ空間における座標
$g(\cdot)$	モジュールの実現する関数
E	評価誤差
k^*	勝者モジュールの番号
$\boldsymbol{w}$	モジュールの結合荷重
$\hat{\boldsymbol{y}}$	モジュールの出力
$\phi(\cdot)$	近傍関数
t	学習ステップ数
$\sigma(t)$	t ステップ目の近傍関数の範囲
ψ	規格化されたモジュールの結合荷重の学習率
σ_0	近傍関数の初期範囲
σ_∞	近傍関数の収束範囲
τ	近傍関数の収束スピード調整パラメータ
η	結合荷重の修正率
β	近傍コピーの安全率

第 1 章

序論

1.1 はじめに

私たちが生きているこの世界では，人間，動物，虫，大気，水そして大地，あらゆるものが絶え間なく動いている．つまり我々のまわりの環境は時々刻々と常に変化しており，この環境下で私たちはおいしいものを食べたり．スポーツを楽しんだり．仕事をしたりといったことをなんの不自由なく行っている．人間だけでなく動物や虫など，“脳” を有する生物はこの常に変化する環境下において柔軟な対応をしながら生活をしている．すなわち脳があるおかげで知的な行動ができるわけである．ではいったい脳とは何なのか．脳とは何でできていて，脳では何が行われているのか．多くの研究者が，“脳” という魅力的な研究素材に対してこのような疑問を持ち，探究心と好奇心をくすぐられ，脳解明のために研究を行っている．そして脳解明には，細胞レベルからのアプローチ，数理モデルからのアプローチ，脳波解析，心理学からのアプローチとさまざまな方面から研究がされている．さらに現在，脳機能の解明に加え，脳の情報処理機能を工学に応用する “脳情報処理工学” という新しい分野ができつつある．この分野ができつつある背景には，これから先最も需要が高くなるであろう最先端医療，宇宙開発などの場面において，マイクロマシンやヒューマノイドロボットなどの人間のように環境の変化に柔軟に対応できる脳情報処理装置を備えたロボットあるいは機器が必須となってくるからである．

さて，過去の生理学的，解剖学的研究によって脳の情報処理の特定の機能は脳の特定の部位と深いかかわりがあるということがわかっている [21]．例えば，脳の後頭部に位置する部位は視覚，頭頂部は運動・感覚，前頭部は感情・思考の情報処理に関わっている．このように，脳は機能ごとに分割でき，分割された領域においてさらに細かい機能ごとに分割されている．つまり機能モジュールが集まったモジュール集合体が脳を形成していると見ることができる．また似た機能モジュールは隣り合う位置に存在するよう規則的に配置されることが知られている．例えば，体の感覚の情報処理に関する部位（大脳皮質運動

野）において，体の隣接する部分（手のひらの隣は腕，親指の隣は人差し指など）に反応する機能モジュールは大脳皮質表面においても隣接するように規則的に配置されていることがPenfieldの実験によって分かっている[39]．このような感覚，機能モジュール間における，脳内機能コラム，あるいは数学的に言われる“トポロジー保存写像”は視覚野や聴覚野においても見られる．そしてこれら“機能モジュールの分散”と“トポロジーを保存した規則的配置”に関連するモデルはいくつか提案されている．それらの中で，Kohonenの自己組織化マップ(Self-Organizing Map : SOM)は脳機能における“トポロジー保存写像”を簡単化した最も代表的なモデルであり，現在SOMは工学的にあらゆる分野で応用されている．一方，“機能モジュールの分散”に関連するモデルとしてはモジュラーネットワークがある．モジュラーネットワークの目的は，大局的な機能を1つのモデルで表現するのではなく，いくつかの局所的な機能に分散させ，局所機能毎にモデルを構築し，全体のモデルを記述することである．

しかし，SOMはトポロジー保存写像は行うが機能モジュールという概念を持っていない．一方，モジュラーネットワークは機能モジュールの規則的配置という概念を持っていない．そこで、これら2つのモデルを融合させれば，脳情報処理モデルとしてさらに工学的に応用できるモデルになるのではないだろうか．このアイデアが着想となったものが本論文である．そこで本論文ではKohonenのSOMとモジュラーネットワークを融合させた“モジュラーネットワークSOM (Modular Network SOM: mnSOM)”を提案し，提案手法であるmnSOMについての研究報告を行う．

1.2 研究の目的・意義

本研究の目的はmnSOMを脳型情報処理の基盤技術として確立することである．しかし，脳における“機能モジュールの分散”と“トポロジー保存”の機能をモデル化したものがmnSOMだ，と単純に言うことはできない．なぜなら脳における情報処理は複雑であり，私が提案したmnSOMのように単純なモデルで記述できるはずがないからである．脳型情報処理を将来実現する際において，従来型の単機能なニューラルネットワークでは明らかに役不足であり，さらにマップを作る以上の機能を持たないSOMでも限界がある．本研究の目指すところは脳の機能や構造に着想を得つつ，知的情報処理を将来可能にするアーキテクチャの基盤を確立することである．従って脳機能の厳密な再現は考えないし，脳機能の解明も目的としない．そこで本論文では，mnSOMのアーキテクチャ，アルゴリズムをシンプルかつ明快な形で提示し，さらにさまざまなシミュレーションによって本提案手法が脳情報処理基盤技術として有効かどうか検証を行う．以下にmnSOMを脳情報処理モデルの基盤技術として確立させる意義を示す．

- **工学的意義**

  環境の変化に柔軟に適応できる制御，多様体上に分布するデータ（顔・表情認識，3D 物体認識など），複雑な時系列データの解析などの課題に対して mnSOM は応用できるだろう．これによって今後需要が大きくなるであろうロボットなどの頭脳（情報処理装置）として mnSOM が適用でき，それによってロボット産業が大きく発達するだけでなく，ロボットが活躍する医療，宇宙開発などの開発にも大きく貢献することが期待できる．

- **脳科学的意義**

  mnSOM は脳内機能コラムの情報処理モデルそのものではないだろうが，脳機能解明の架け橋となると期待できる．また複雑時系列処理の開発によって脳機能解明を手助けするだろうと期待できる．

  なお mnSOM は特定のアーキテクチャを目指すのではなく，SOM とモジュラーネットワークを融合した広い概念を指す．したがって mnSOM は，一種のアーキテクチャ群のファミリーを指す言葉であり，従来型 SOM もファミリーの一員である．すなわち mnSOM は SOM を一般化した概念である．

1.3 本論文の構成

先にも述べたように，本論文では mnSOM に対して脳情報処理基盤技術の確立という視点から研究を行っている．そのため本論文は生理学的な知見から脳機能における mnSOM の位置付けをすることではなく，情報処理工学の視点から論文は書かれている．

それでは以下に本論文の構成をまとめる．まず mnSOM は Kohonen の SOM と古川の競合モジュラーネットワーク [11] を基に提案されたものである．ゆえに，2 章の基礎知識でこれら 2 つの手法について説明をする．3 章では mnSOM ファミリーの最も代表的なメンバーである多層パーセプトロン (Multi-Layer Perceptron : MLP) をモジュールとする mnSOM，すなわち MLP-mnSOM について取り上げ，そのアーキテクチャアルゴリズムについて説明する．4 章では MLP-mnSOM のマップの特性を見出すために簡単な 3 次関数集合を用いたシミュレーションについて述べる．このシミュレーションによって mnSOM の生成するマップの特性は通常の SOM と変わらないことが示される．5 章では，MLP-mnSOM の応用として気象時系列データ集合を用いたシミュレーションを示す．ここでは九州 20 都市の気象データを用いて mnSOM でマップを生成した．その結果，mnSOM は気象時系列データ集合に対して都市の地理的位置のトポロジーを保存したマップを生成した．6，7 章では mnSOM の課題に対する汎化性の検証が示される．まず 6 章では，mnSOM のモジュールがリカレントニューラルネットワーク (Recurrent Neural

Network : RNN) で構成された RNN-mnSOM が，ダイナミカルシステム集合に対して応用ができることを示す．このことは mnSOM が適応制御，複雑時系列データ集合の解析などに対して応用可能であることを意味する．7 章では，mnSOM のモジュールが自己想起型ニューラルネットワーク（Autoassociative Neural Network : ANN）で構成された ANN-mnSOM が Nonlinear ASSOM を実現するかの検証結果を示す．この検証によって ANN-mnSOM が多様体集合に対してマップを生成でき，その特徴は顔認識，表情認識，3D 物体認識などに応用できることが示唆された．8 章は考察・討論である．mnSOM をさまざまな課題に応用する時に mnSOM が構成されるモジュールのデザインをどのように決定すればよいのか，mnSOM を実現する場合の注意点，一般化された mnSOM などについて示す．

第 2 章

基礎知識

2.1 Self-Organizing Map

Kohonen の Self-Organizng Map (SOM) は多次元ベクトルデータを位相保存的に低次元の特徴空間へ写像するアルゴリズムである．したがって高次元空間のデータ同士の関係を視覚的に捉えやすい地図（マップ）として表現することができ，データ解析ツールの一つとしてポピュラーな手法である [24]．さらに SOM はアーキテクチャやアルゴリズムがシンプルなため工学的応用が容易であり，医療 [33, 31]，通信 [6, 22]，画像・音声の解析・認識 [13, 27, 19]，制御 [34, 36]，ロボット工学 [49, 14]，経済 [4]，遺伝子解析 [20, 1] など幅広い分野にわたって SOM は応用されてきた．また生理学的な面から SOM を見ると脳内における感覚・運動機能の自己組織的配置過程に似ている [24]．このことは SOM を基本として脳機能モデルを組み立てることができるのではないかと考えられる．以上の点より，SOM は非常に興味深い手法である．

本節では SOM のアーキテクチャおよびアルゴリズムを説明する．

2.1.1 アーキテクチャ

SOM のアーキテクチャを図 2.1 に示す．SOM の構造は非常にシンプルで，SOM は単一ベクトルを実現するユニットが 1 次元あるいは 2 次元状に配置された構造を持ち，各ユニットの位置は固定されている（ユニットが 1 次元状に配置されている場合は 1 次元 SOM，2 次元の場合は 2 次元 SOM と呼ぶことにする）．SOM には多数の特徴から成る複数のベクトル，つまり多次元特徴空間上の複数のベクトルが入力される（図 2.2）．そして SOM の競合作用である勝者丸取り (Winner-Take-All : WTA) によって，SOM の各ユニットは特徴空間を局所空間に分割する代表ベクトルを実現する．さらに SOM の協調作用（近傍関数）によって，隣り合うユニットは隣り合う局所空間の代表ベクトルを実現する．

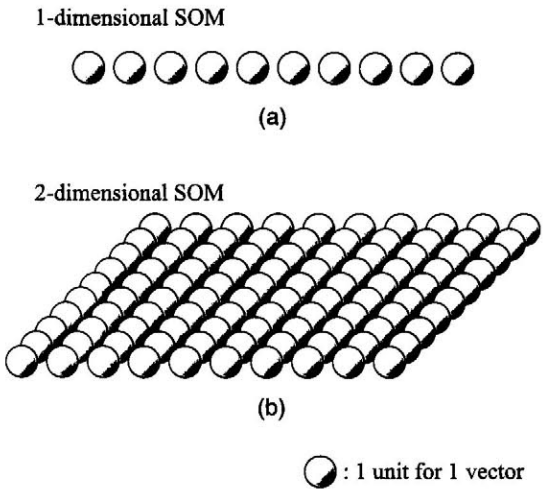


図 2.1 SOM のアーキテクチャ．1 個の球がユニットを表し，1 ユニットは 1 ベクトルだけを実現する．(a) 1 次元 SOM．(b) 2 次元 SOM．

すなわち，SOM は入力された特徴空間から SOM 全体のユニットで表現されるマップ空間へトポロジーを保存して写像を行う（図 2.2）．以下に SOM について詳しく説明する．

今，d 個の特徴から成る d 次元ベクトル $\boldsymbol{x}_i = (x_{i1}, x_{i2}, ..., x_{id})^T$ が I 個 ($X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_i, ..., \boldsymbol{x}_I\}$) あるとする．このとき各ベクトル $\boldsymbol{x}_1, ..., \boldsymbol{x}_i, ..., \boldsymbol{x}_I$ は，$x_{i1}, x_{i2}, ..., x_{id}$ を特徴ベクトルとする d 次元特徴空間 $\mathcal{X}$ 上に点在する（図 2.2）．また SOM は K 個のユニットから構成され，これらユニットが並んでいる空間をマップ空間 $\mathcal{A}$ とする．さらに k-th ユニットが実現するベクトルを $\boldsymbol{m}^k$，k-th ユニットのマップ空間における座標を $\boldsymbol{\xi}^k$（1 次元 SOM の場合：$\boldsymbol{\xi}^k = \xi_1^k$，2 次元 SOM の場合：$\boldsymbol{\xi}^k = \left(\xi_1^k, \xi_2^k\right)$ とする．SOM には $\boldsymbol{x}_1, ..., \boldsymbol{x}_i, ..., \boldsymbol{x}_I$ が入力され，SOM における競合作用と協調作用によって特徴空間 $\mathcal{X}$ からマップ空間 $\mathcal{A}$ へトポロジーを保存して写像が行われる．以下に競合作用と協調作用について説明する．

(1) 競合作用 (WTA)

SOM において WTA だけが作用した場合，SOM は入力特徴空間に対してベクトル量子化 (Vector Quantization:VQ) [2, 12] をすることに等しい．VQ は入力特徴空間を複数の局所空間に分割し，各局所空間内に属するベクトルは類似しているとみなして，おのおのの局所空間に代表ベクトル（コードベクトル）を持たせることによって，入力特徴空間全体を少ない代表ベクトルで近似（つまりデータ圧縮) することである*1．つまり，空

*1 VQ の代表的な手法として k-means 法 [32] や LVQ(Learning Vector Quantization) [24] がある．k-means 法は教師なし学習，LVQ は教師あり学習である．また VQ で分割される局所空間の境界は各代表ベクトルを中心とする超球面同士が交差する点を結んだ超平面となる．この分割をボロノイテセーション

間 $\mathcal{X}$ を局所空間 $\mathcal{X}_{d1}, ...\mathcal{X}_{dK}$ に分割し，各局所空間を SOM のユニットが実現するベクトル $\boldsymbol{m}^1, ..., \boldsymbol{m}^K$ で量子化することになる（図 2.3）．このとき，VQ の目的は以下に示す量子化誤差の二乗の期待値 E が最小になるようにコードベクトル $\boldsymbol{m}^k$ を決定することである [28].

$$E = \int_X \|\boldsymbol{x} - \boldsymbol{m}^{k_i^*}\|^2 p(\boldsymbol{x})\, d\boldsymbol{x} \tag{2.1}$$

ここで $p(\boldsymbol{x})$ は$\boldsymbol{x}$の確率密度関数（*p.d.f.*）である．さらに $\boldsymbol{m}^{k_i^*}$ は入力ベクトル $\boldsymbol{x}_i$ に最も類似したコードベクトル[*2]を示し，$\boldsymbol{m}^{k_i^*}$ を実現する k_i^* ユニットを勝者ユニット（または Best Matching Unit : BMU）と呼ぶ．式 2.1 の最小化問題は

$$\boldsymbol{m}^{k_i^*} \leftarrow \boldsymbol{m}^{k_i^*} + \alpha\left(\boldsymbol{x}_i - \boldsymbol{m}^{k_i^*}\right) \tag{2.2}$$

の繰り返し計算で近似的に解けることが分かっている [24]．ここで α は修正量を表し通常 $0 < \alpha \leq 1.0$ とする．以上より，SOM において競合作用のみが働いた場合は，入力ベクトルに対して勝者ユニットのコードベクトルのみが修正を許されることがわかる（図 2.4(a)）．なお，SOM アルゴリズムにおいて近傍関数なしで展開する位相マップと VQ との関係は Luttrell [30, 29] によって確立されている．

以上より SOM において競合作用だけが働いた場合，勝者ユニットの近傍のユニットが特長空間において距離の近いコードベクトルを実現せず，特徴空間のトポロジーを保存したマップは生成できない．ゆえに勝者ユニットだけでなく近傍のユニットもコードベクトルの修正を許す協調作用が必要となる．

(2) 協調作用（近傍関数）

SOM は競合作用だけではく協調作用も同時に働く．ゆえに式 2.1 は以下の式に書き換えられる [29].

$$E = \frac{1}{2}\sum_{i=1}^{I}\sum_{k=1}^{K}\phi_i^k(\boldsymbol{\xi}^{k_i^*}, \boldsymbol{\xi}^k)\|\boldsymbol{x}_i - \boldsymbol{m}^k\|^2. \tag{2.3}$$

ここで $\phi_i^k(\boldsymbol{\xi}^{k_i^*}, \boldsymbol{\xi}^k)$ は近傍関数であり，WTA のように勝者ユニットだけを修正するのではなく，勝者の近傍のユニットも似たコードベクトルを実現するように修正する働きがある（図 2.4(b)）．このときの修正量は近傍関数で定義され，勝者ユニットから離れるにつれて修正量も小さくなる（図 2.5）．通常，近傍関数としては次式のようなガウス関数が用

（Voronoi tesselation），分割された局所空間をボロノイ空間（またはボロノイ領域）と呼ぶ．

[*2] 式 2.1 より，入力ベクトルとコードベクトルの類似度はユークリッド距離で定義される．一般的に SOM はユークリッド空間において類似度を測るため類似度にユークリッド距離が定義されるが，距離の定義を変えることによって SOM が生成するマップも異なる．

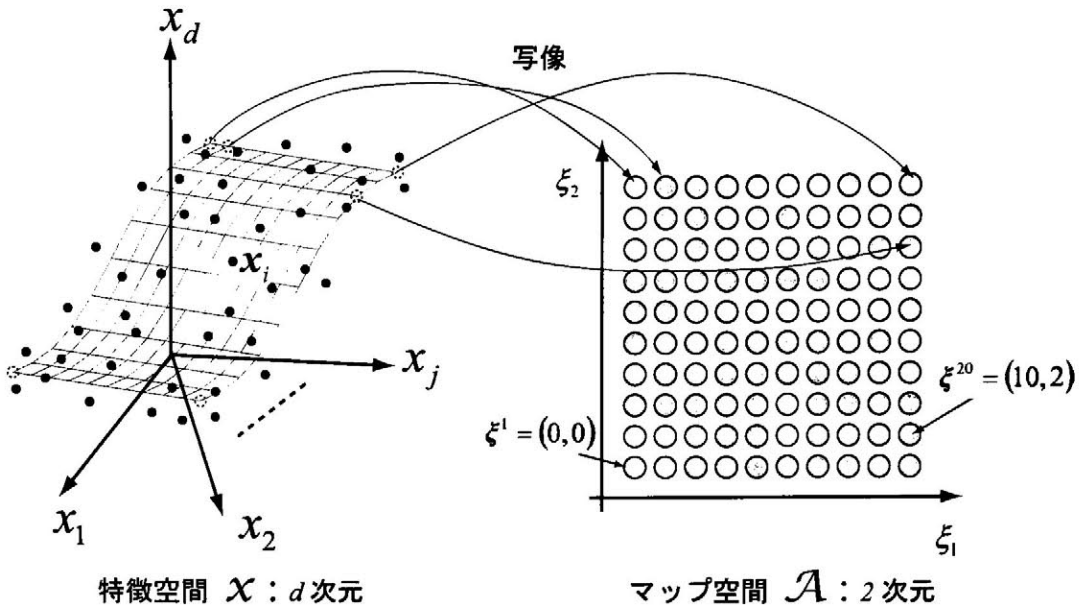


図 2.2 特徴空間 $\mathcal{X}$ からマップ空間 $\mathcal{A}$ への写像

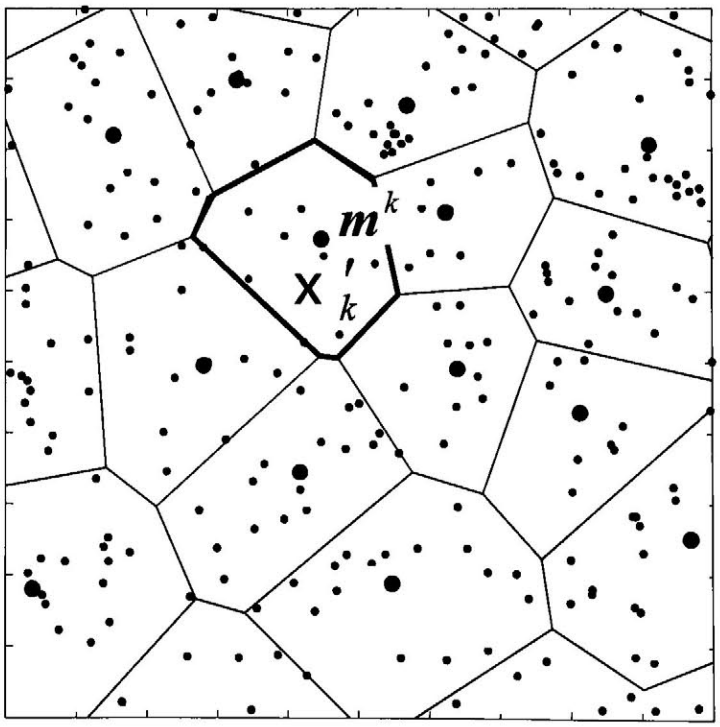


図 2.3 コードベクトルによって分割された特徴空間の一例：各コードベクトルによって量子化される局所空間の境界は区分的に超平面となりボロノイゼーションを形成する.

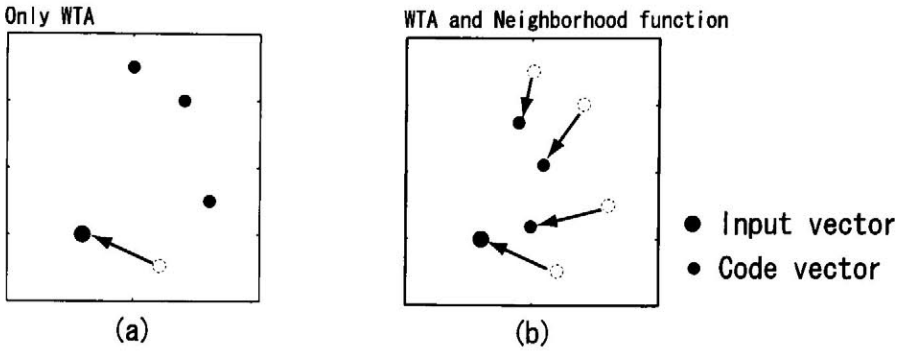


図 2.4 VQ(WTA のみ作用), SOM におけるコードベクトルの動き. (a) 競合作用である WTA だけが働く場合, 勝者ユニットだけがコードベクトルの更新を許される. (b) 競合作用 (WTA) と協調作用 (近傍関数) が同時に働く SOM の場合, 勝者ユニットだけでなく近傍のユニットもコードベクトルの更新を許される.

いられる.

$$\phi_i^k(\boldsymbol{\xi}^{k_i^*}, \boldsymbol{\xi}^k) = \exp\left(-\frac{\|\boldsymbol{\xi}^{k_i^*} - \boldsymbol{\xi}^k\|^2}{2\sigma^2}\right). \tag{2.4}$$

σ は近傍関数の広さを決める定数であり, 通常, SOM の学習回数 t の関数 (単調減少関数) とし, 学習を進めるにつれて近傍の広さを狭めていく (図 2.6). 以下 $\phi_i^k(\boldsymbol{\xi}^{k_i^*}, \boldsymbol{\xi}^k)$ は ϕ_i^k だけを表記する. SOM の目的は式 (2.3) を評価関数として, E 最小化するようにコードベクトルを修正することである.

さて, コードベクトルの更新法には大別して二通りの方法がある. ひとつは, オンライン型と呼ばれるものであり, 他方はバッチ型と呼ばれるものである. 両者の本質的な違いは近傍関数の和が各コードベクトルごとに 1 になるように規格化するかどうかの違いであり, 逐次的か一括的かは本質ではない. そこで本論文では誤解を避けるため, オンライン型をナイーブ SOM, バッチ型をベイズ SOM と呼ぶことにする (これらの呼称は本論文のみのものであり, 一般に使われているわけではない. しかし「逐次型バッチ SOM」のような矛盾した表現を避けるため, 本論文ではこの呼称を用いる). 逐次的な学習はナイーブ SOM, ベイズ SOM のどちらでも可能であり, 一方, 一括型はベイズ SOM でのみ可能である. なお, ベイズ SOM による一括学習はナイーブ SOM による逐次学習と比べはるかに高速であり, また安定した計算結果をもたらす. この比較については 2.1.3 説で詳しく述べる.

ベイズ SOM

ベイズ SOM のアルゴリズムはいくつかの方法で理論的に導出が可能である. ひとつは一般化ロイドアルゴリズムからの導出である [30, 29]. ノイズを含むベクトル量子化にお

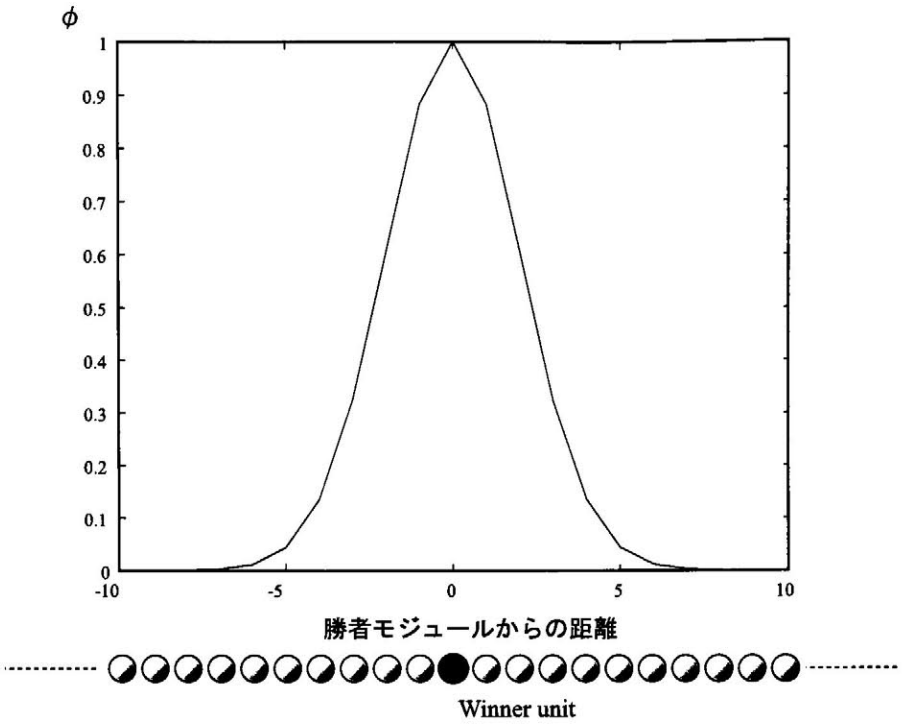


図 2.5 近傍関数の修正量．勝者ユニットは修正量が 1.0 であり，勝者から離れるにつれて修正量は近傍関数（図はガウス関数の場合）に従って減少する．

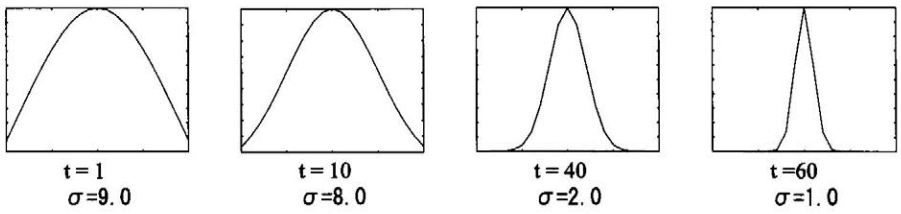


図 2.6 近傍関数のスケジューリング．学習回数が増えるにともなって，近傍関数の広さが狭まるようにする．

いて，量子化誤差を最小とするアルゴリズムを計算するとベイズ SOM の更新式に帰着する．すなわち，ベイズ SOM はデータ点に確率的不確定性がある条件下での量子化誤差最小化問題を解いているといえる．

ベイズ SOM のもう一つの導出は期待値最大化 (EM) アルゴリズムを用いた方法であり，Bishop は Generative Topographical Map (GTM) として理論的に示した [3]．また Furukawa らは近傍関数の規格化がベイズ則に相当することを示し，同じく EM アルゴリズムを使って一般化 SOM のアルゴリズムを導出した [18, 43]．

ベイズ SOM の利点はいくつかある．一つは各コードベクトルがデータベクトルの局所

的線形内分点に収束することが保証されていることである．このため安定したマップが生成される．また各コードベクトルの学習速度は一定であり，コードベクトルの更新係数を大きくとることができる．そのため学習速度が速い．学習係数をもっとも大きくした状態が一括学習である．

ナイーブ SOM

ベイズ SOM と異なり近傍関数の計算結果をそのまま学習量に反映させたものがナイーブ SOM である．Kohonen が当初提案した SOM はナイーブ型であった [24]．その後，Kohonen は LBG アルゴリズム [28] を元に「バッチマップ」を提案したが，これがベイズ SOM に相当する．理論的に見ればナイーブ SOM はベイズ SOM の近似表現であり，各コードベクトルにおける近傍関数で獲得した学習量の総和が，コードベクトルごとに大きな差がないという仮定の下で学習量の規格化を省略することに当たる．このような状況はデータベクトル数が十分に大きいときに成立する．

ナイーブ SOM にはいくつかの欠点があり，それはしばしばマップ生成の失敗につながる．

1. コードベクトルがデータベクトルの局所内分点に収束することが保証されない．そのため近傍関数のスケジューリングによって結果が左右される．
2. コードベクトルの初期値に対する依存性が大きく，初期状態が悪いと適切なマップが作られない．
3. 学習の速度がコードベクトルごとに異なる．そのため学習速度の最も遅いユニットに学習をあわせる必要があり，その結果，膨大な繰り返し計算が必要になる．一般にベイズ SOM による一括学習が数十回の繰り返しで収束するのに対し，ナイーブ SOM の逐次学習だと数千回程度の繰り返しが必要になる．またコードベクトルごとの学習速度の差がマップ生成の失敗につながることもある．ナイーブ SOM による一括学習はできない．
4. 近傍関数の他に学習係数が必要であり，学習係数は近傍関数とは異なるスケジューリングで減少させる必要がある．このため，スケジューリングが二重に必要になるため，学習パラメータの決定に手間を要する．また学習パラメータに対する結果の依存性も大きい．これはパラメータの変動に対してロバストなベイズ SOM と対照的である．

以上のようにナイーブ SOM は多くの欠点を持つが，歴史的には先に提案されたことと，アルゴリズムが簡便なことから今でも利用する人は少なくない．

2.1.2 アルゴリズム

SOMのアルゴリズムは大きく分けて，*Evaluative process*，*Competitive process*，*Cooperative process*，*Adaptive process*の4つのプロセスに分けられる．この4つのプロセスを繰り返すことがSOMの学習アルゴリズムである．以下に各プロセスの詳細を示す．

Evaluative process

*Evaluative process*では，入力ベクトル$\boldsymbol{x}_i$と各ユニットが実現しているコードベクトル$\boldsymbol{m}^k$のユークリッド距離（類似度）E_i^kを評価する．

$$E_i^k = \|\boldsymbol{x}_i - \boldsymbol{m}^k\| \qquad {}^\forall k \tag{2.5}$$

このとき，ナイーブSOMの場合は，1つの入力ベクトル$\boldsymbol{x}_i$をランダムにピックアップするので，$\boldsymbol{x}_i$だけについて評価を行う．一方，ベイズSOMの場合は，全ての入力$\boldsymbol{x}_i \qquad (i = 1, ..., I)$について評価を行う．

Competitive process

*Competitive process*では，入力ベクトル$\boldsymbol{x}_i$に対して最も類似度E_i^kの小さいユニットを勝者ユニットk_i^*に決定する．つまり，

$$k_i^* = \arg\min_k E_i^k \tag{2.6}$$

となる．

Cooperative process

*Cooperative process*では，近傍関数を用いて各ユニットが実現するコードベクトル$\boldsymbol{m}^k$の修正量を決定する．このとき，ナイーブSOMの場合は，近傍関数として$\phi_i^k(\boldsymbol{\xi}^{k_i^*}, \boldsymbol{\xi}^k)$を用いる．一方，ベイズSOMの場合は次式によって規格化された近傍関数ψ_i^kを用いる．

$$\psi_i^k = \frac{\phi_i^k}{\sum_{i=1}^I \phi_i^k} \tag{2.7}$$

Adaptive process

*Adaptive process*では*Cooperative process*で求めた修正量に従って各コードベクトルを更新する．つまり逐次更新をするナイーブSOMの場合は，

$$\boldsymbol{m}^k(t+1) = \boldsymbol{m}^k(t) + \lambda(t)\phi_i^k\|\boldsymbol{x}_i - \boldsymbol{m}^k(t)\|, \tag{2.8}$$

一括更新をするベイズ SOM の場合は，

$$\boldsymbol{m}^k = \sum_{i=1}^{I} \psi_i^k \boldsymbol{x}_i \tag{2.9}$$

となる．

なお，以上の 4 プロセスを行う前に，初期設定を行う必要がある．初期設定を行う必要があるものは主に，

- 各ユニットが実現するコードベクトル $\boldsymbol{m}$
- 近傍関数のスケジューリングの設定
- マップのサイズ（ユニットの数）

である．各項目について最適な設定はなく，課題に応じて設定を変える必要がある．とくにナイーブ SOM の場合，初期値に依存しやすいので注意が必要である．

2.1.3 計算機シミュレーション

SOM の特徴を示すために (1) 正方格子内に一様に分布するデータ (2) 動物データ，の 2 つのデータに対するそれぞれの計算機シミュレーションを示す．

(1) 正方格子内に一様に分布するデータ

正方格子に囲まれた空間内において入力ベクトルが一様に分布しているデータに対して，オンライン型ナイーブ SOM とバッチ型ベイズ SOM のそれぞれで学習を行った．結果の一例を図 2.7 に示す．ナイーブ（図 2.7(a)），ベイズ（図 2.7(b)）の両方とも $t = 1$ における初期状態から徐々に SOM のマップが広がっていく様子がわかる．しかし，両手法においてマップの生成される過程と最終的に生成されたマップには違いがある．ナイーブ SOM の結果ではマップの生成に時間を要し，さらに生成されたマップは歪んでおり正方空間全体にコードベクトルが広がっていないが，ベイズ SOM では，マップ生成の時間が短くさらに生成されたマップは入力空間である正方空間全体に広がっている．このような両手法における違いは以下のように考察できる．ナイーブ SOM の場合，1 入力ベクトルに対して全体のコードベクトルが移動するのでコードベクトルは入力があるたびに動き回る．従って，マップの生成過程は初期値，入力ベクトルを与える順番，それから近傍関数のスケジュールにとても敏感である．また，時間を掛けてゆっくりとマップを形成しなければ正方空間全体にコードベクトルが行き届かない．一方，ベイズ SOM は全体の入力ベクトルに対して全コードベクトルの移動する位置が一意的に決まるために初期値や近傍関数のスケジュールの設定に鈍感でありマップの生成は安定している．さらに入力空間を均等に分割しようとする Bayse の特長により正方空間全体にコードベクトルは広がる．こ

こで図2.8にNaiveとBayseの両者が生成したマップにおけるボロノイ分割を示す．ベイズSOMのコードベクトルに対するボロノイ空間は綺麗な正方格子を示しており，入力空間をSOMのコードベクトルで均等に分割している．これより，ベイズSOMはナイーブSOMより安定したマップ生成が行えることがわかる．

(2) 動物データ

このシミュレーションでは表2.1に示す動物データをSOMに与え，動物個体間の関係をSOMのマップ上に表現することを目的とする．ここでライオン，チータ，ハト，等の動物の種類が入力ベクトルであり，ここでi番目の動物の特徴ベクトルを$\boldsymbol{x}_i = (x_{i1}, x_{i2}, ..., x_{iJ})$とする．ベクトルの各要素は「夜行性」や「泳ぐ」などのその動物の特徴になる．つまり17個のベクトルが21次元特徴空間に分布し，この入力ベクトルに対してSOMを用いマップを生成する．なお本シミュレーションはバッチ型ベイズSOMで行った．

結果を図2.9に示す．各格子はSOMのユニットを示し，格子内に書かれている文字は対応するユニットが勝者となる動物名である．また格子内の色は，各ユニットが隣接するユニットに比べてどれくらい似たベクトルを実現しているかを示す指標をカラーバー色で表している．青であるほど隣接ユニットとの類似度が高い．すなわち，k-thユニットに隣接するユニットの集合をHとし，実現されるベクトルを$\boldsymbol{m}^h \quad h \in H$とするならば，$k$-thユニットの色$Cr^k$は以下の式で決定される．

$$Cr^k = \frac{1}{|H|} \sum_{h \in H} \|\boldsymbol{m}^k - \boldsymbol{m}^h\| \tag{2.10}$$

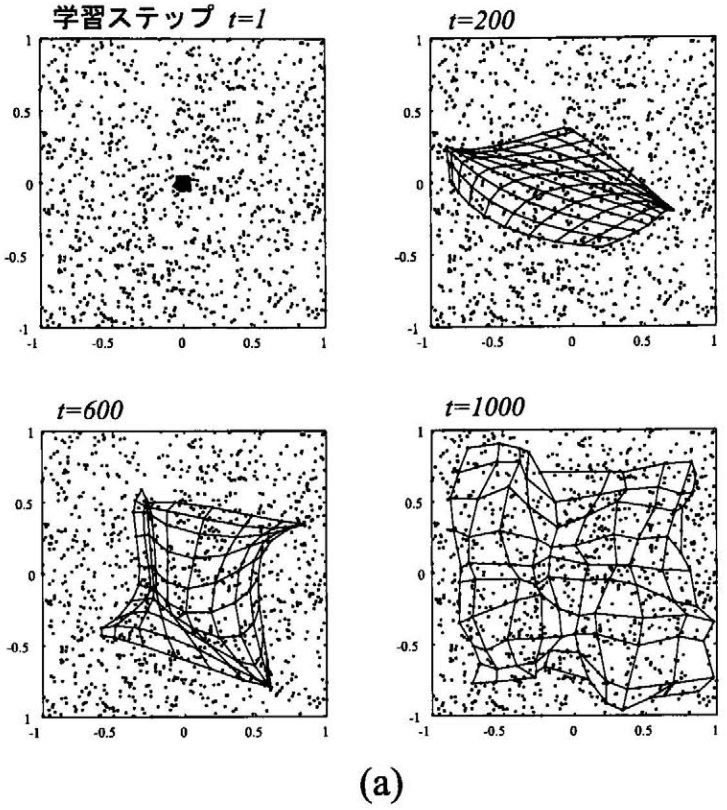
Cr^kの値が大きければ$k-th$ユニットと隣接するユニットは類似性がなく，色は赤で表示される．一方Cr^kの値が小さければ類似性があり，色は青で表示される．

さて，結果を見ると，マップの中央部が赤系の色になった．そしてマップ中央より右側には鳥類，左側には哺乳類がそれぞれマップされた．つまり，マップの中央部に境界線が引かれていることになる．またマップの上部は草食系，下部は肉食系の動物がそれぞれマップされた．

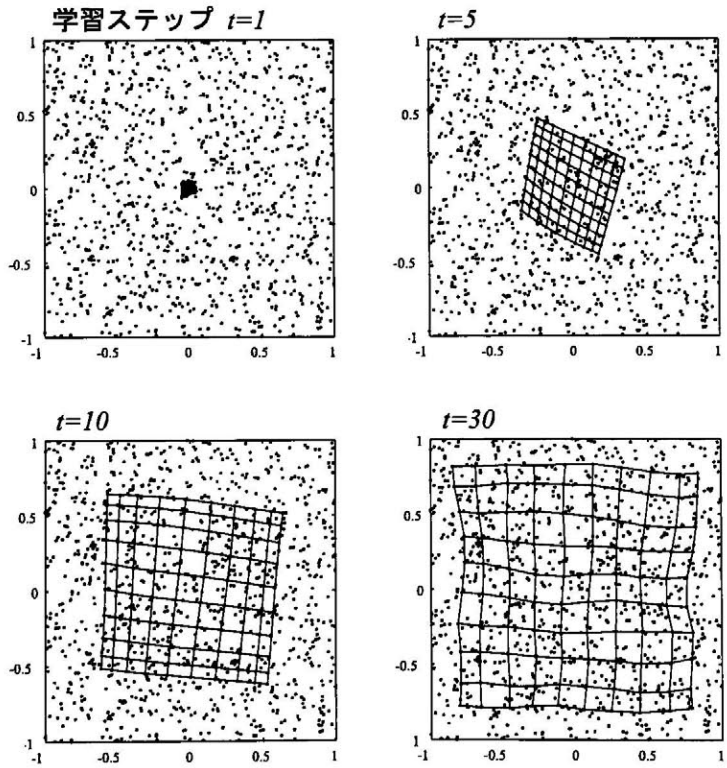
このように，SOMは多次元ベクトルデータに対してベクトル間の関係を我々が視覚的に捉えることのできる地図（マップ）として表すことができる．

2.2 モジュラーネットワーク

SOMは特徴空間を分割し，分割された空間をコードベクトルで代表させる．このときSOMは分割されたどの空間においても，データベクトルの密度がほぼ同じになるようにコードベクトルを配置する．そして分割された各領域におけるベクトルは同一クラスに属することになる（つまりクラスタリングをする）．しかし，ある連続して分布しているベ



(a)



(b)

図 2.7 ナイーブ，ベイズ SOM の比較結果:(a) ナイーブ SOM. (b) ベイズ SOM.

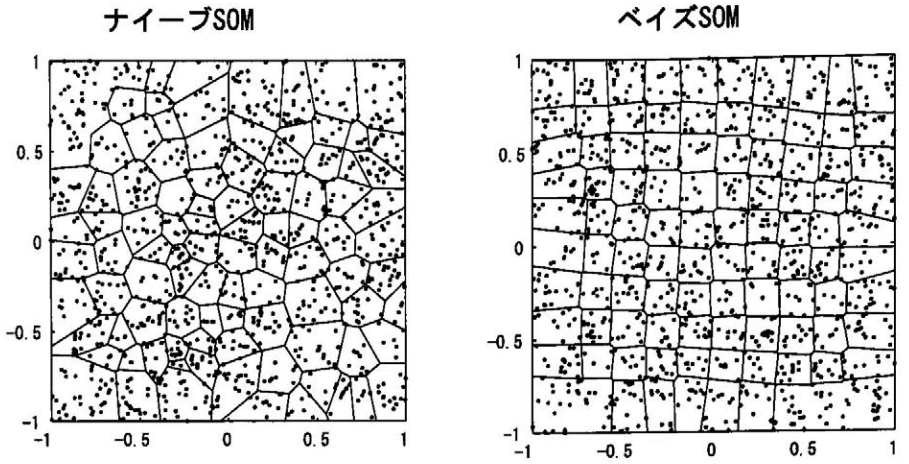


図 2.8 ナイーブ，ベイズ SOM のボロノイ空間比較結果 (a) ナイーブ SOM. (b) ベイズ SOM.

表 2.1 動物データ

	ハト	メンドリ	アヒル	カモ	カラス	フクロウ	タカ	ワシ	キツネ	イヌ	オオカミ	ネコ	トラ	ライオン	ウマ	シマウマ	ウシ
小型	1	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
中型	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
大型	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
夜行性	0	0	0	0	0	1	0	0	0.5	0	1	0.5	0.5	0	0	0	0
本足	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
本足	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
毛を持つ	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
蹄を持つ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
鬣を持つ	0	1	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
を持つ	1	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
縞がある	0	0	0	0.3	0	0	0	0	0	0	0	0	1	0	0	1	0
狩猟を好む	0	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
速く走る	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
空を飛べる	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
泳ぐ	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
家畜	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1
草食	1	1	1	1	0.5	0	0	0	0	0.5	0	0.5	0	0	1	1	1
肉食	0	0	0	0	0.5	1	1	1	1	0.5	1	0.5	1	1	0	0	0
イヌ科	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
ネコ科	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
ペット	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0

クトル集合を 1 クラスとしてクラスタリングをして欲しいときに，SOM のような VQ をベースとするアルゴリズムでは，1 つのクラスを複数の空間に分割（つまり複数のクラスに分割）するだろう．

ここで視点を変えると，SOM のコードベクトルは特徴空間の分割された領域におけるモデル（平均値）であると解釈もできる．このように特徴空間をいくつかの部分空間に分割し，各部分空間におけるモデルを獲得する手法として，モジュラーネットワークがある．代表的なモジュラーネットワークとして Jacobs と Jordan の Mixture of Exparts モデルが

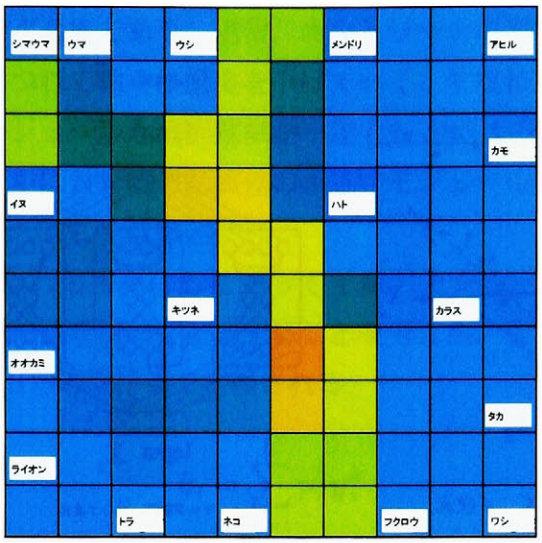


図 2.9 SOM による動物データのマップ

あげられる [15, 17]. Mixture of Expart モデルは特徴空間全体をモデル化する場合に，1つのモデルで特徴空間全体を記述すると複雑なモデルになるので，特徴空間を適当な領域に分割し，局所的かつ単純なモデルの組み合わせでモデル全体を記述することができる．また古川はデータ空間の最適な分割が，与えられたデータのクラスリングにもつながるという視点から，与えられたデータ集合を局所的なサブモデルの組み合わせで表現することを通し，最適なモデル化と最適なクラスタリングを同時に行うモジュラーネットワークを提案した [11].

このように，視点を変えることによって SOM とモジュラーネットワークは，特徴空間を適当な領域に分割し各局所空間においてモデルを獲得するという類似している点を持つ．これが本論文の提案手法であるモジュラーネットワーク SOM に発展する．本章ではモジュラーネットワーク SOM の基となる古川の提案した競合モジュラーネットワークについて説明する．

2.2.1 競合モジュラーネットワークのアーキテクチャアルゴリズム

古川の競合モジュラーネットワークのアーキテクチャを図 2.10 に示す．競合モジュラーネットワークにおいて各モジュールは 3 層砂時計型 MLP (Multi-Layer Perceptron), あるいは 3 Layer Autoassociative Neural Netowork (3L-ANN) と呼ばれるニューラルネットワークで構成される．3L-ANN は入力層と出力層のユニットの数 N_D が同じであり，隠れ層のユニット数 N_H は入出力層のユニットより少ない数で構成されたバックプロパゲーション型 MLP である．3L-ANN のタスクは入出力層より少ない数の隠れユニットで

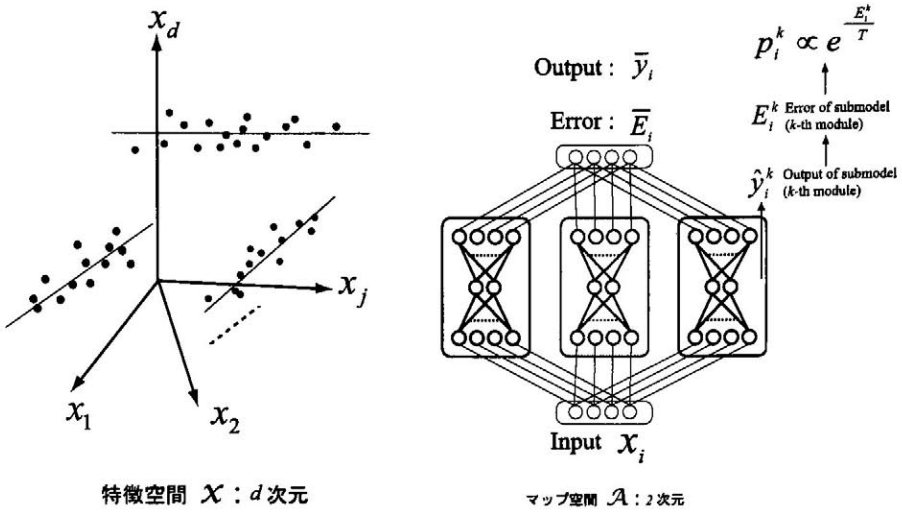


図 2.10 競合モジュラーネットワークのアーキテクチャ

入力ベクトルを出力層で想起することである．つまり 3L-ANN の教師信号は入力ベクトルと等しい．このとき，入力ベクトルの情報は隠れユニットで圧縮された後に出力で入力ベクトルを想起するので，学習を行うと隠れユニットには入力ベクトル空間の基底ベクトルが発現される．すなわち 3L-ANN は入力ベクトルに対して主成分分析をすることに等しい[*3]．競合モジュラーネットワークのタスクは入力ベクトル $\boldsymbol{x}$ に対して想起誤差（モデル誤差）が小さいモジュールほど高確率で入力ベクトルの担当モジュール k^*（あるいは勝者モジュールと呼ぶ）に選択され，さらに，k^* は $\boldsymbol{x}$ に対してモデル誤差が小さくなるように学習を許される．これによって競合モジュラーネットワークはモデル化とクラスタリングを同時に行うことが可能となる．

以下に具体的に説明する．今，I 個の入力ベクトル $\{\boldsymbol{x}_i\}$ が特徴空間 $\mathcal{X}$ において分布しているとする．またモジュールの数を K とする．このとき競合モジュラーネットワークの全てのモジュールに $\boldsymbol{x}_i$ が入力され，各 3L-ANN モジュールの出力 $\boldsymbol{y}_i$ との想起誤差 E_i^k が計算される．このとき，E_i^k は次式で定義される．

$$E_i^k = \frac{1}{2}\|\boldsymbol{x}_i - \boldsymbol{y}_i\|^2 \tag{2.11}$$

そして $\boldsymbol{x}_i$ に対して k-thMLP が選択される確率 p_i^k は次式で定義される．

$$p_i^k = \frac{\exp\left[-E_i^k/T\right]}{\sum_{k'}\exp\left[-E_i^{k'}/T\right]} \tag{2.12}$$

[*3] 5 層砂時計型 MLP（5L-ANN）の場合は非線形主成分分析を実現する．このことについては 6 章で説明する．

ここで，T はアニーリング温度であり，通常，ネットワークの学習初期段階では高い温度に設定し，学習が進むにつれて徐々に温度を下げていく．すなわち，学習の初期段階では温度が高いため，どのモジュールも選択確率が一定であるが，学習の進行につれて温度が下がるため，入力ベクトルに対する想起誤差の小さいモジュールが入力ベクトルに特化して選択されるようになる（詳しくは後述）．

さて，式 2.12 で与えられる確率にしたがって 1 個のモジュールを選択する代わりに，ネットワークの動作を期待値を用いて記述することを考える．まずネットワーク全体の出力値 $\hat{\boldsymbol{y}}_i$ は次式のように出力の期待値で定義される．

$$\hat{\boldsymbol{y}}_i = \sum_{k=1}^{K} p_i^k \boldsymbol{y}_i^k \tag{2.13}$$

同じようにネットワーク全体の誤差も，誤差の期待値 $\bar{E}_i$ で次式のように定義する．

$$\bar{E}_i = \sum_{k=1}^{K} p_i^k E_i^k \tag{2.14}$$

そして $\bar{E}_i$ が最小になるようネットワークを学習させる．つまり各 3L-ANN モジュールはバックプロパゲーション法 (BP) によって学習されるので k-th モジュールの結合荷重 $\boldsymbol{w}^k$ の更新量は以下の式で更新される．

$$\triangle \boldsymbol{w}^k = -\eta \frac{\partial \bar{E}_i}{\partial \boldsymbol{w}^k} \tag{2.15}$$

$$= -\eta \left\{ p_i^k \frac{\partial E_i^k}{\partial \boldsymbol{w}^k} + \sum_{k'=1}^{K} E_i^k \frac{\partial p_i^{k'}}{\partial \boldsymbol{w}^k} \right\} \tag{2.16}$$

$$= -\eta \, \psi_i^k \, \frac{\partial E_i^k}{\partial \boldsymbol{w}^k} \tag{2.17}$$

すなわち，上式は通常の BP の結合荷重更新の式に ψ_i^k を乗じたものであり，i-th データに対して k-th モジュールがどのくらい学習するかを決める学習率である．学習率 ψ_i^k は次式で与えられる．

$$\psi_i^k = p_i^k \left\{ 1 + \frac{1}{T} \left(\bar{E}_i - E_i^k \right) \right\} \tag{2.18}$$

この学習率は次のことを意味する．(1) 勝者モジュールは他のモジュールより多く学習をする．(2) 勝者モジュールと他のモジュールの誤差が近い場合は，勝者モジュールは学習率が 1 を超え，その他のモジュールは学習率が負になり（つまり反学習），両者の差が開くように学習は行われる．(3) 温度 T が小さい時は誤差の大小に関わらず学習が進む（結果的にどの MLP もほぼ均等に学習する）．温度が小さくなるにつれて勝者モジュールのみが学習するようになる．したがって温度 T を徐々に下げながら学習を行うと，特定の

データ集合に適したモジュールが現れ，データのクラスタリングが自己組織的に行われることが期待できる．

以上が古川の提案した競合モジュラーネットワークのアーキテクチャ・アルゴリズムである．このようなモジュラーネットワークを用いた手法は他にもさまざまあり，代表的なものとして Jacobs と Jordan の Mixture of Exparts モデルが挙げられる [15, 17]．このモジュラーネットワークは複数のエキスパートネットワークと呼ばれる複数のモジュールとゲーティングネットワークと呼ばれるモジュールで構成される．そして各モジュールの学習率はゲーティングネットワークの出力値で決まる．つまり入力データに対するモジュールの選択はゲーティングネットワークによってコントロールされている．一方，各モジュールの誤差を用いてネットワークの学習率を決定するモデルとして，Wolpert らのモデル [48] がある．Wolpert らのモデルはクラス分類を目的としたものではないが，古川のモデルと基本的には等価なネットワークを持つ．しかしこのモデルでは，各モジュールの学習率は各モジュールの選択される確率 p_i^k をそのまま用いており，これは T が一定下で用いられた場合に相当する．

第 3 章

mnSOM のアーキテクチャとアルゴリズム

3.1 はじめに

Kohonen の SOM は教師なし学習法の 1 つであり，高次元空間のベクトルデータを 1, 2 次元程度の低次元空間へトポロジーを保存しつつ写像することにより，ベクトルデータ間の関係を視覚的に捉えられる地図（マップ）を生成することができる [24]．これより SOM はデータ分類，データ解析といったデータマイニングツールとして利用されることが多い [16, 47]．さらに SOM はネットワークの構造や理論がシンプルであるため工学的にも応用が簡単であり様々な分野で広く使われている．一方，応用だけでなく SOM のアーキテクチャやアルゴリズムを拡張した手法も多く提案されてきた．しかし，今日まで扱われてきた SOM は，ほとんどの場合，SOM のユニットがベクトルデータしか扱えず，ベクトル空間においてマップを生成される原理は残ったままである．

これに対し，本論文で私は従来型 SOM におけるベクトルユニットをニューラルネットワークなどのモジュールに置き換えたモジュラーネットワーク SOM (modular network SOM : mnSOM) を提案する．mnSOM はモジュラーネットワークの学習に SOM の基本原理である競合・協調作用を導入したものと考えることができ，各モジュールはコードベクトルだけではなくさまざまな '機能' を実現することができる．すなわち，mnSOM は，ただ単に SOM を拡張したものではなく，さまざまなデータタイプに適用できるような一般化された SOM を実現する．もし mnSOM が SOM としての特徴を持ち合わせつつ，時系列データ集合，システム（関数）集合などのさまざまなデータタイプに適用できるならば，工学的にも非常に効果的なツールとなるだろう．また，過去の生理学的，解剖学的研究によって，脳における特定の情報処理機能は脳の特定の部位と深い関わりがあることがわかっており，これは脳が機能モジュールの集合体であることを裏付けている [21]．さら

にPenfieldによって各機能は大脳皮質上において類似性に基づいて整列していることがわかっている[39]. mnSOMは脳からヒントを得て考案されたものではないが，結果的に大脳皮質の機能モジュールとよく類似した構造を持つ．したがって，脳の情報処理メカニズムを工学的に実現する土台としても有効かもしれない．

まず私はmnSOMを提案しアーキテクチャとアルゴリズムを確立した．本章では，mnSOMのモジュールをMLPとした場合（MLP-mnSOM）のアーキテクチャとアルゴリズムについて説明する．一般化されたmnSOMについては第8章の考察で述べる．なお，本章の内容は文献[46]で報告されている．

3.2 アーキテクチャおよび本研究における問題の枠組み

mnSOMのアーキテクチャを図3.1に示す．mnSOMは，通常のSOMにおける各ユニットをニューラルネットワークなどの機能モジュールに置き換えたモジュラーネットワークの構造を持つ．そして各モジュールの学習としてSOMにおける競合作用(WTA)と協調作用(近傍関数)を導入する．mnSOMの機能モジュールは，mnSOMの対象となる課題に合わせて自由にデザインすることができる．本論文では多層パーセプトロン(Malti-Layer Perceptron : MLP)，Elmanのリカレントニューラルネットワーク(Recurrent Neural Network : RNN) [5]や自己想起型ニューラルネットワーク(Autoassociative Neural Network : ANN) [37]などを用いてシミュレーションを行っているが，本章においては機能モジュールがMLPの場合についてのみ説明をする．なぜなら，本論文を読む読者にとって，機能モジュールをMLPとした場合が一番mnSOMの特徴を理解しやすいからである．なお本稿では以降，MLPを機能モジュールとするmnSOMを以後MLP-mnSOMと呼ぶ．一般化された場合のアーキテクチャについては，第8章の考察で述べる．

まず本研究におけるmnSOMが扱う問題設定について説明する．図3.2に示すようにMLP-mnSOMはK個のMLPモジュール$M^1, ..., M^K$で構成され，学習に用いるデータはI個の関数（システム）から観測された入出力データであるものとする．ただしmnSOMにはサンプリングデータのみが与えられ，関数形は未知とする．この状況でmnSOMは以下のように動作することを目的とする．

1. I個の入出力データ集合をK個のモジュールを使って学習し，元となる関数形を学習により獲得・再現する．K個のモジュールのうちI個がそれぞれの関数に対する勝者モジュールとなる．関数形の獲得は教師あり学習である．
2. 勝者以外のモジュールは学習によって中間的な関数形を獲得し，関数と関数の間を補間しながら，全体として自己組織化マップが生成される．
3. このとき関数同士の類似・相違度を表す指標として関数空間における距離を用い

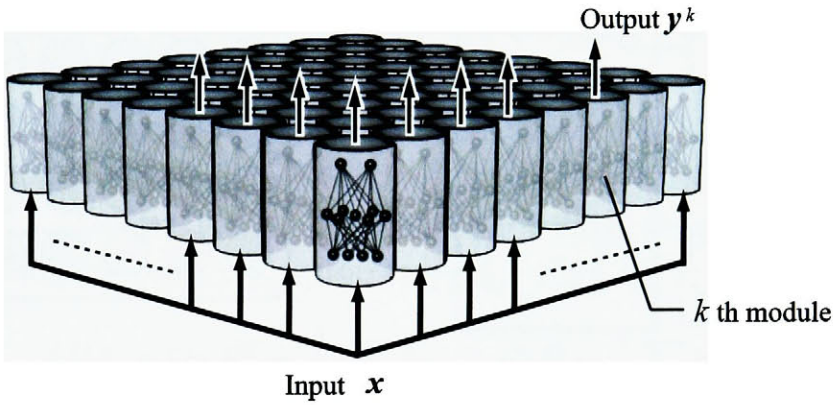


図 3.1 mnSOM のアーキテクチャ．図は MLP を機能モジュールとする mnSOM を示す．

る．このマップの生成は教師なし学習によって生成される．

これらを定式化すると次のようになる．今，$\boldsymbol{x}$ を d_{in} 次元入力ベクトル，$\boldsymbol{y}$ を d_{out} 次元出力ベクトルとする I 個の関数（システム）があり，各システムから J 個の入出力データのペアからなる観測データ集合 $D_i = \{(\boldsymbol{x}_{ij}, \boldsymbol{y}_{ij})\}$ $(i = 1, ..., I\ ;\ j = 1, ..., J)$ が得られているとする．すなわち

$$\begin{aligned} D_i &= \{(\boldsymbol{x}_{ij}, \boldsymbol{y}_{ij})\} \\ \boldsymbol{x}_{ij} &= [x_{ij1}, ..., x_{ijd_{in}}]^T \\ \boldsymbol{y}_{ij} &= [y_{ij1}, ..., y_{ijd_{out}}]^T \end{aligned} \tag{3.1}$$

である．また各関数は

$$\boldsymbol{y} = f_i(\boldsymbol{x}) = f_i(\boldsymbol{x}; \boldsymbol{\theta}_i) \tag{3.2}$$

と記述されるとする．ここで $\boldsymbol{\theta}_i$ は $f_i(\cdot)$ を決めるパラメータベクトルである．そして $f_i(\cdot)$ がどんな関数であるかは事前にわからず，その代わりランダムにサンプルされた入出力データの集合 D_i が得られているとする．

mnSOM の第一の目的は，これら $I \times J$ 個の入出力データから，$f_1(\cdot), ..., f_I(\cdot)$ を学習により獲得することである．これらは I 個の勝者モジュールによって実現されることが期待される．mnSOM の第二の目的は，上記 I 個以外のモジュールを用いて，中間的な関数形を持つ関数を学習し，関数間を補間することである．そして第三の目的は，I 個の関数

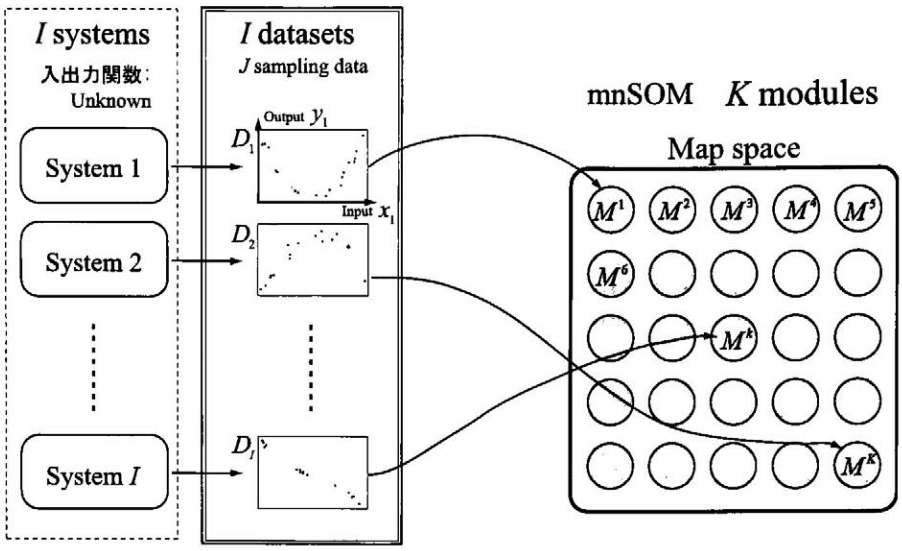


図 3.2 本研究における mnSOM が扱う問題設定.

$f_1(\cdot), ..., f_I(\cdot)$ 同士の関係をマップ上へ写像することである．関数 f, g の類似度は関数空間における距離,

$$L^2(f,g) = \int \|f(\boldsymbol{x}) - g(\boldsymbol{x})\|^2 p(\boldsymbol{x}) d\boldsymbol{x} \tag{3.3}$$

によって定義される.

なお $p(\boldsymbol{x})$ はサンプリング時の $\boldsymbol{x}$ の確率密度関数である．ここで，入力ベクトルの確率密度関数 $p(\boldsymbol{x})$ はクラスに依存せず一定である必要があり，一定でなければ距離 (3.3) が定義できない.. このためクラスごとに $p(\boldsymbol{x})$ が異なる場合には距離を定義できなくなるが，その場合は $\{\boldsymbol{x}\}$ の平均が 0，分散が 1 になるように事前にデータを規格化し，$p(\boldsymbol{x})$ がクラスに依存せず，ほぼ一致するように前処理を施しておく．なお $p(\boldsymbol{x})$ がクラスごとに異なる場合については 8． 4 で考察する.

3.3 アルゴリズム

次に MLP-mnSOM のアルゴリズムについて説明する．一般的に mnSOM のアルゴリズムは SOM と同じように *Evaluative process*, *Competitive process*, *Cooperative process*, *Adaptive process* の 4 つのプロセスで構成されている．これら 4 プロセスを一回の学習として，繰り返し学習を行う.

Evaluative process

Evaluative process では，各モジュール $\{M^k\}$ の実現する関数 $\{g^k(\cdot)\}$ と全クラスの関数 $\{f_i(\cdot)\}$ との距離 $L(f_i, g^k)$ を式 3.3 で評価する．しかし実際には各クラスの関数 $\boldsymbol{y} = f(\boldsymbol{x})$ は未知なので，式 3.3 を用いて直接評価することはできない．そこで，各関数から得られたデータセット $\{D_i\}$ を用いて近似的に評価を行う．

まず i-th 関数における j-th 入力データ $\boldsymbol{x}_{ij}$ を全モジュールに入力し，対応する出力 $\hat{\boldsymbol{y}}_{ij}^k = g^k(\boldsymbol{x}_{ij})$ を求め，k-th モジュールの出力 $\hat{\boldsymbol{y}}_{ij}^k$ と望ましい出力（教師信号）$\boldsymbol{y}_{ij}$ の自乗誤差 $e_{ij}^k = \|\boldsymbol{y}_{ij} - \hat{\boldsymbol{y}}_{ij}^k\|^2$ を計算する．以上の過程を全クラス全データに対して行い，e_{ij}^k のアンサンブル平均 E_i^k で式 3.3 を近似的に評価する．すなわち，

$$L^2(f_i, g_k) \simeq E_i^k = \frac{1}{2J} \sum_j \|\boldsymbol{y}_{ij} - \hat{\boldsymbol{y}}_{ij}^k\|^2 = \frac{1}{J} \sum_j e_{ij}^k \tag{3.4}$$

となる．

Competitive process

Competitive process では，全クラスの関数に対してそれぞれ勝者モジュールを決める．つまり次式に示すように i-th 関数に対して，最も評価誤差 E_i^k が小さかったモジュールを勝者モジュール k_i^* とする．すなわち

$$k_i^* = \arg\min_k E_i^k \tag{3.5}$$

となる．

Cooperative process

Cooperative process では，各モジュールの学習量を求める．i-th 関数に対する k-th モジュールの学習量 ψ_i^k は以下の式で求められる．

$$\psi_i^k = \frac{\phi_i^k\left(d(k, k_i^*); t\right)}{\sum_{i'} \phi_{i'}^k\left(d(k, k_{i'}^*); t\right)} \tag{3.6}$$

ここで，$\phi(\cdot; t)$ は学習ステップ t における近傍関数を示し，$d(k, k_i^*)$ はマップ空間における勝者モジュール k_i^* と k-th モジュールとの距離を示す．本研究では近傍関数 $\phi(\cdot)$ は以下に示す式を用いる．

$$\phi(d; t) = \exp\left[-\frac{d^2}{2\sigma^2(t)}\right] \tag{3.7}$$

ここで $\sigma(t)$ は近傍関数の広さを調整する関数であり，通常 $\sigma(t)$ は学習の始まりは大きく設定し，学習が進むにつれ減少する単調減少関数が用いられる．なお本研究では $\sigma(t)$ の

スケジューリングを次式のように与えた.

$$\sigma(t) = \sigma_\infty + (\sigma_0 - \sigma_\infty) \exp\left(-\frac{t}{\tau}\right). \tag{3.8}$$

ここで学習初期（$t = 0$）の近傍関数の広さを σ_0，$t = \infty$ としたときに近傍関数が収束する広さを σ_∞，τ を近傍関数のスケジューリング係数とする.

Adaptive process

Adaptive process では，全てのモジュールを次式のように誤差逆伝播法 (Backpropagation 法 : BP 法) によって学習させる.

$$\Delta \boldsymbol{w}^k = -\eta \sum_{i=1}^{I} \psi_i^k \frac{\partial E_i^k}{\partial \boldsymbol{w}^k} = -\eta \frac{\partial E^k}{\partial \boldsymbol{w}^k}. \tag{3.9}$$

ここで，$\boldsymbol{w}^k$ は k-th モジュールの結合荷重を表している．また $E^k = \sum_i \psi_i^k E_i^k$ である．つまり，E^k は各モジュールの誤差エネルギーであり，この誤差エネルギーが大域的最小値となる部分が各モジュールの実現する関数 g^k となる．このとき g^k は次式に示すように $\{f_i\}$ の重み付線形内分点（重心）となる.

$$g^k(\boldsymbol{x}) = \sum_{i=1}^{I} \psi_i^k f_i(\boldsymbol{x}). \tag{3.10}$$

以上のアルゴリズムはベイズ SOM が基本となっている（第2章を参照）．そのため，もし厳密にバッチ型ベイズ SOM のアルゴリズムに従うならば，式 3.4 は $\left\{g^k(\cdot)\right\}$ が収束するまで繰り返すことになる．逆にオンライン型のアルゴリズムに従うならば，式 3.4 は 1 回だけすれば十分ということになる．すなわち式 3.4 を繰り返す回数は任意でよい．現実的には，細かく勝者判定をすると $\boldsymbol{w}^k$ がゆっくりとしか変化しないので，式 3.4 を十分繰り返したほうが効率が良い.

第 4 章

多項式関数集合を用いた mnSOM の有効性の検討

本章では，MLP-mnSOM が期待する性能を持つことを多項式関数集合を用いたシミュレーションによって示す．また，MLP-mnSOM が関数空間における SOM を実現することも証明する．

4.1 シミュレーションの枠組み

シミュレーションで用いた 3 次関数を図 4.1 に示す．これらの 3 次関数は，

$$y = ax^3 + bx^2 + cx \tag{4.1}$$

の a, b, c をさまざまに設定した 126 個の関数である．本シミュレーションは，二つのケースについて行われる．まず “ケース A” では，図 4.1 中の太枠で囲まれた 6 個（つまり $I = 6$）の関数を学習用クラスとして用いる（それ以外は用いない）．このとき，各クラスのサンプルデータ数は 200 個とする（$J = 200$）．一方 “ケース B” では，図 4.1 中に示す 126 個（$I = 126$）の関数全てを学習用クラスとして用いる．このとき，各クラスのサンプルデータ数は 8 個とする（$J = 8$）．どちらのケースにおいても，各クラスのデータセット $D_i = \{(x_{ij}, y_{ij})\}$ は，$[-1.0, 1.0]$ の間で確率密度関数 $p(x)$ が一定となるようにランダムにサンプリングされる．さらに，出力 y_{ij} には正規白色雑音（ガウシアンホワイトノイズ（ガウシアンの分散 $\sigma_{noize} = 0.04$）が加わっている．図 4.2 にシミュレーションで用いたいくつかのデータセットの例を示す．図 4.2(a)，(b)，(c) はケース A で用いられたデータセットで，(d)，(e)，(f) はケース B で用いられたデータセットである．

ケース A では，与えられたデータセットから各クラスの関数を推定することは簡単である．また与えられた関数間の中間的な関数を発現することも期待できる．一方，ケース

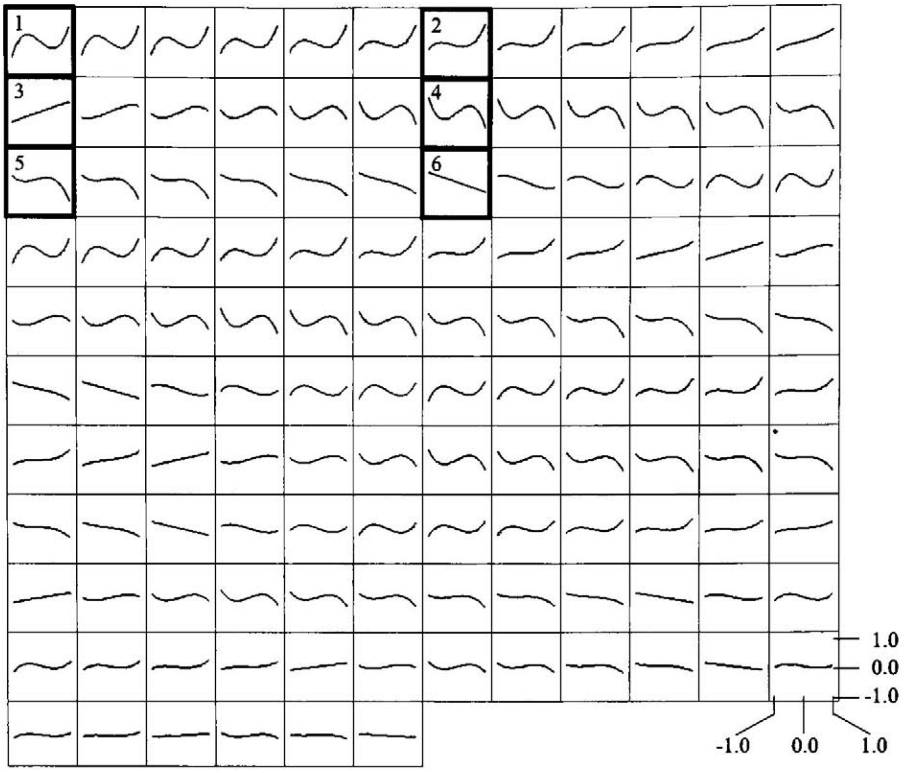


図 4.1 シミュレーションで用いる 126 個の 3 次関数形

B では不完全なデータセットが mnSOM には与えられるので，各クラスの関数を推定することは困難である．しかし，mnSOM は近傍のクラスを手懸りとして，126 個の関数を推定すると期待できる．また mnSOM は図 4.3 に示すように，入力層：1 ユニット，出力層：1 ユニット，隠れ層：8 ユニットで構成された 3 層 MLP を機能モジュールとする．他の詳細は表 4.1 に示す．

4.2 シミュレーションの結果

図 4.4(a), (b) にそれぞれケース A，B の結果を示す．各格子は mnSOM のモジュールに対応しており，格子内の曲線は対応するモジュールが実現した関数形を示す．まず，どちらのケースとも mnSOM は似たマップを生成した．つまり，隣り合うモジュールは似た関数形を実現し，性質の異なる関数同士は互いにマップ上で対角となる場所に位置した．またケース A（図 4.4(a)）では，学習で与えられた関数は 6 個と少ないが，これら 6 関数の勝者モジュール（図 4.4(a)）中の太枠で囲まれたモジュール）以外のモジュールは関数間を内挿するような関数を実現した．一方，ケース B の結果（図 4.4(b)）では，少ない数のデータベクトルで構成された学習データセットを用いたにもかかわらず，3 次関数集合

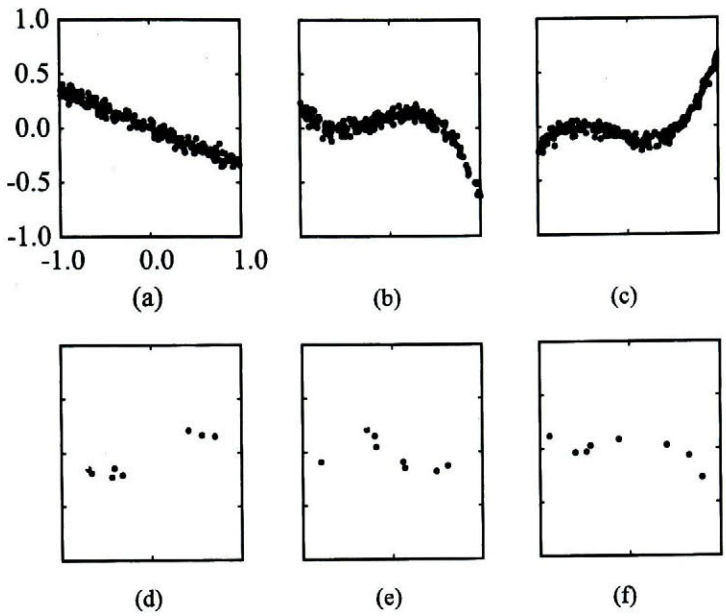


図 4.2 データセットの例：(a),(b),(c) はケース A で用いたデータセット，(d),(e),(f) はケース B で用いたデータセット

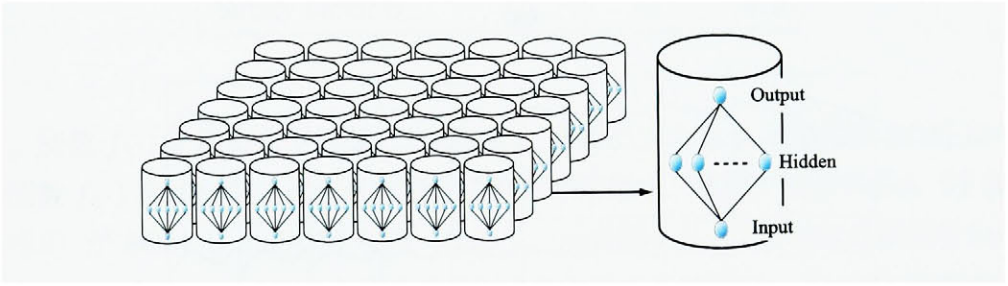


図 4.3 3 次関数集合の自己組織化マップを生成する MLP-mnSOM のアーキテクチャ

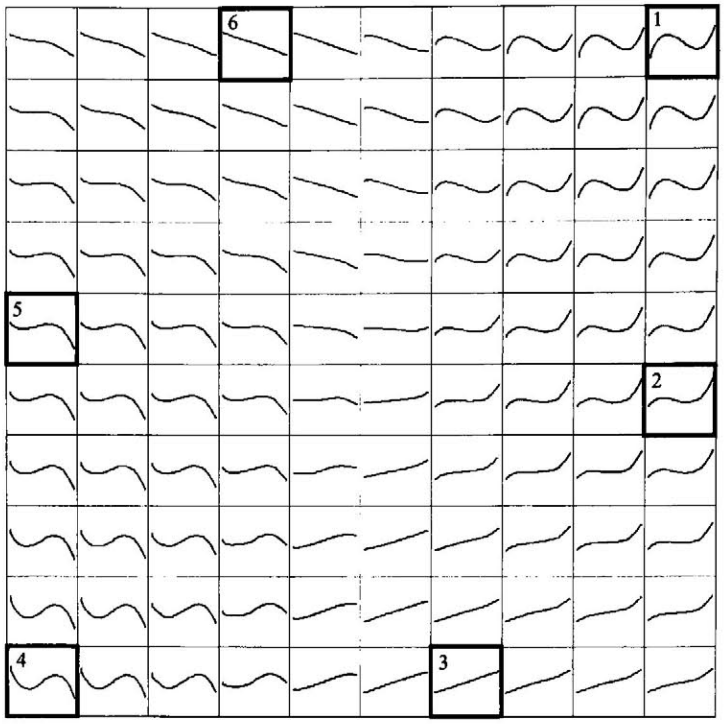
のマップ生成は成功したといえる.

4.3 考察

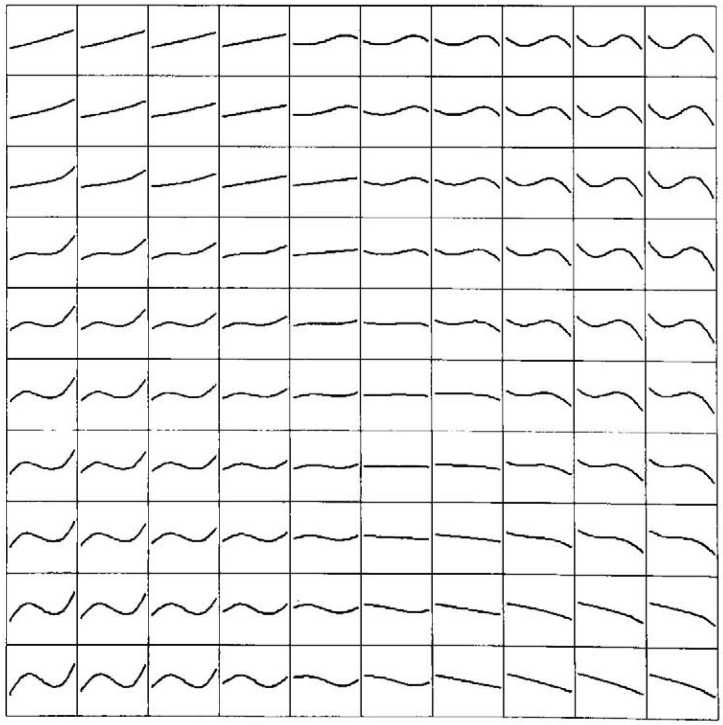
シミュレーションでは，mnSOM は与えられたデータセットから関数を獲得するだけでなく，同時に関数の自己組織化マップを生成した．これより mnSOM は期待通りの性能を有すると言えるだろう.

さて，関数 $f(\cdot)$ は正規直交展開をすることによって，次式のように基底関数 $\{P_n(\cdot)\}$ と係数ベクトル $\boldsymbol{a} = (a_1, a_2, ..., a_n)$ から成る関数となる．このとき係数ベクトル空間と関数空間の両方における距離は理論的に一致する（これは証明 1 に示す).

$$f(x) = a_1 P_1(x) + a_2 P_2(x) + ... + a_n P_n(x) \tag{4.2}$$



(a)



(b)

図 4.4 MLP-mnSOM による 3 次関数集合のマップ. (a) ケース A の結果. (b) ケース B の結果.

表 4.1 3 次関数集合を用いたシミュレーションにおける mnSOM の各種パラメータ

Parameters of MLP module	
Number of units	
Input layer	1
Hidden layer	8
Output layer	1
Learning constant η	0.05
Parameters of mnSOM	
Map size K	100 (10×10)
Neighborhood function size	
σ_0	10.0
σ_∞ (case 1)	2.0
σ_∞ (case 2)	1.0
τ	300
Safety factor β	0.9

つまり，関数 $f(\cdot)$ を正規直交展開したのちに，係数ベクトル $\boldsymbol{a}$ を従来の SOM に与えた場合と関数 $f(\cdot)$ を mnSOM に与えた場合の両方の結果は，学習アルゴリズム（すなわち，式 3.4- 3.9）が SOM も mnSOM も同一なので，mnSOM と SOM は同じ結果を生成するべきである．つまり，mnSOM と従来の SOM の生成する特徴マップは同じ特性を有しているはずである．この事実は，従来の SOM による数々の仕事が MLP-mnSOM でも導入できるということを意味する．

以上を検証するために，本論文では関数の正規直交展開として Legendre 展開を用いた．本シミュレーションにおいて，図 4.1 に示す 126 個の関数は Legendre 多項式の 3 次項までを用いた関数から作られたものである．つまり，mnSOM に与えられた関数 $\{f_i\}$ は，

$$f_i(x) = a_{i1}P_1(x) + a_{i2}P_2(x) + a_{i3}P_3(x) \tag{4.3}$$

$$\begin{aligned} P_1 &= \sqrt{7}\left(\frac{5}{2}x^3 - \frac{3}{2}x\right) \\ P_2 &= \sqrt{5}\left(\frac{3}{2}x^2 - \frac{1}{2}\right) \\ P_3 &= \sqrt{3}x \end{aligned} \tag{4.4}$$

の係数 $\boldsymbol{a}_i = \{(a_{i1}, a_{i2}, a_{i3})\}$ をさまざまに変えた関数集合である．図 4.1 中の太枠で囲ま

表4.2 ケースAのシミュレーションで用いた6関数のLegendre多項式の係数

$a_{11} = 0.4$	$a_{12} = 0.4$	$a_{13} = 0.4$
$a_{21} = -0.4$	$a_{22} = -0.4$	$a_{23} = -0.4$
$a_{31} = 0.2$	$a_{32} = 0$	$a_{33} = 0$
$a_{41} = -0.2$	$a_{42} = 0$	$a_{43} = 0$
$a_{51} = 0$	$a_{52} = 0$	$a_{53} = 0.2$
$a_{61} = 0$	$a_{62} = 0$	$a_{63} = -0.2$

れた6個の関数の係数 $\boldsymbol{a}$ を表4.2に示す.

この $\boldsymbol{a}_1, ..., \boldsymbol{a}_6$ の係数ベクトルをSOMに与えて生成されたマップと，本シミュレーションのケースAで生成されたマップが等しければmnSOMとSOMの生成するマップの特性が等しいといえる．図4.5(a)にSOMによって生成された係数ベクトル $\{\boldsymbol{a}_i\}$ のマップを示す．この図はSOMの各ユニットが実現したコードベクトルを係数ベクトル空間上にプロットしたものである．またmnSOMによって生成されたマップの場合を図4.5(b), (c)に示す．図4.5(b)は式4.5によって評価される各MLPモジュールのグローバルミニマムポイントをプロットしたマップである．つまり，次式で示される $\{\boldsymbol{b}^k\}$ をプロットしたものである.

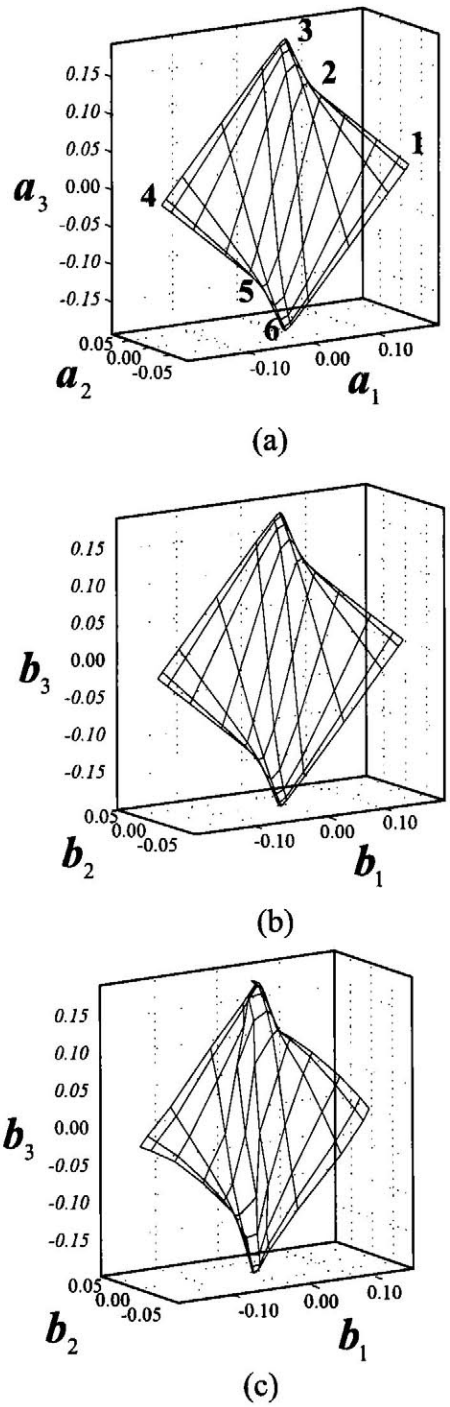
$$
\begin{aligned}
g^k(x) &= \psi_1^k f_1(x) + \psi_2^k f_2(x) + ... + \psi_6^k f_6(x) \\
&= \{a_{11}P_1(x) + a_{12}P_2(x) + a_{13}P_3(x)\}\,\psi_1^k \\
&\quad + \{a_{21}P_1(x) + a_{22}P_2(x) + a_{23}P_3(x)\}\,\psi_2^k \\
&\quad + ... + \{a_{I1}P_1(x) + a_{I2}P_2(x) + a_{I3}P_3(x)\}\,\psi_I^k \\
&= \left\{a_{11}\psi_1^k + a_{21}\psi_2^k + ... + a_{I1}\psi_I^k\right\} P_1(x) \\
&\quad + \left\{a_{12}\psi_1^k + a_{22}\psi_2^k + ... + a_{I2}\psi_I^k\right\} P_2(x) \\
&\quad + \left\{a_{13}\psi_1^k + a_{23}\psi_2^k + ... + a_{I3}\psi_I^k\right\} P_3(x) \\
b_1^k &= \left\{a_{11}\psi_1^k + a_{21}\psi_2^k + ... + a_{I1}\psi_I^k\right\} \\
b_2^k &= \left\{a_{12}\psi_1^k + a_{22}\psi_2^k + ... + a_{I2}\psi_I^k\right\} \\
b_3^k &= \left\{a_{13}\psi_1^k + a_{23}\psi_2^k + ... + a_{I3}\psi_I^k\right\}
\end{aligned}
\tag{4.5}
$$

一方，図4.5(c)はmnSOMの各MLPモジュールによって獲得された関数から次式によっ

て Legendre 多項式の係数 $\{\boldsymbol{b}^k\}$ を求めてプロットしたものである．

$$
\begin{aligned}
b_1^k &= g^k(x)P_1(x) \\
b_2^k &= g^k(x)P_2(x) \\
b_3^k &= g^k(x)P_3(x)
\end{aligned}
\tag{4.6}
$$

これらのマップを見ると，図 4.5(a), (b), (c) はどれもおおむね一致していることがわかる．すなわち，mnSOM と SOM の生成するマップの特性は等しいといえる．つまり MLP-mnSOM は関数空間における SOM ということが理論的にも証明された．なお，図 4.5(c) のマップは，図 4.5(a) と比べて歪みが生じている．これは mnSOM のモジュールで使用している MLP の関数の実現精度が悪いからである．関数の実現精度を上げることによって，図 4.5(c) の歪みは消えるだろう．このためには MLP の中間層ユニットの数を増やしたり，MLP をモジュールとするのではなく，Radial Basic Function (RBF) などをモジュールとするべきだろう．



(a)

(b)

(c)

図 4.5 SOM と mnSOM のマップ特性検証結果. (a)SOM のマップ. (b)mnSOM の各 MLP モジュールのグローバルミニマムポイント. (c)mnSOM のマップ.

証明 1

正規直交関数 $f(\boldsymbol{x})$ と $g(\boldsymbol{x})$ があるとする．このとき $f(\boldsymbol{x}) = a_1P_1(\boldsymbol{x}) + ... + a_nP_n(\boldsymbol{x})$ と $g(\boldsymbol{x}) = w_1P_1(\boldsymbol{x}) + ... + w_nP_n(\boldsymbol{x})$ との 2 乗距離 $L^2(f(\boldsymbol{x}), g(\boldsymbol{x}))$ は以下のように定義される．

$$L^2(f(\boldsymbol{x}), g(\boldsymbol{x})) = \int \|f(\boldsymbol{x}) - g(\boldsymbol{x})\|^2 p(\boldsymbol{x})d\boldsymbol{x} \tag{4.7}$$

これは以下のように展開できる．

$$\begin{aligned}
L^2(f(\boldsymbol{x}), g(\boldsymbol{x})) &= \int \|f(\boldsymbol{x}) - g(\boldsymbol{x})\|^2 p(\boldsymbol{x})d\boldsymbol{x} \\
&= \int \|(a_1P_1(\boldsymbol{x}) + ... + a_nP_n(\boldsymbol{x})) - (w_1P_1(\boldsymbol{x}) + ... + w_nP_n(\boldsymbol{x}))\|^2 p(\boldsymbol{x})d\boldsymbol{x} \\
&= \int \|(a_1 - w_1)P_1(\boldsymbol{x}) + ... + (a_n - w_n)P_n(\boldsymbol{x})\|^2 p(\boldsymbol{x})d\boldsymbol{x}
\end{aligned}$$

ここで $P_i(\boldsymbol{x})$ と $P_j(\boldsymbol{x})$ の積を P_{ij} と表記する．

$$= \int \left((a_1 - w_1)^2 P_{11}(\boldsymbol{x}) + 2(a_1 - w_1)(a_2 - w_2)P_{12}(\boldsymbol{x}) + ...\right) p(\boldsymbol{x})d\boldsymbol{x}$$

$\{P_i\}$ は直交しているので，$P_{ij} = 0$，$P_{ii} = 1$ となる．すなわち，

$$\begin{aligned}
&= \int \left((a_1 - w_1)^2 + (a_2 - w_2)^2 + ... + (a_n - w_n)^2\right) p(\boldsymbol{x})d\boldsymbol{x} \\
&= \sum_{i=1}^{n} (a_i - w_i)^2 \int p(\boldsymbol{x})d\boldsymbol{x} \\
&= \sum_{i=1}^{n} (a_i - w_i)^2 = L^2(\boldsymbol{a}, \boldsymbol{w}) \\
L^2(f(\boldsymbol{x}), g(\boldsymbol{x})) &= L^2(\boldsymbol{a}, \boldsymbol{w})
\end{aligned} \tag{4.8}$$

以上より正規直交関数を基底とする関数空間の距離はベクトル空間の距離と等しくなる．

第5章

mnSOMによる気象ダイナミクスの自己組織化マップ

前章で用いたデータは人工データだったが，本章ではmnSOMの応用として全国各地の気象データを用いた計算機シミュレーションについて述べる．さらに本章では基本モジュールとしてMLPとSOMにおけるベクトルユニットをハイブリッドしたアーキテクチャを採用しており，基本モジュールを課題にあわせて変えることによって従来のSOMでは不可能であったマップ生成もできることを示している．

5.1 はじめに

日本各地の気象ダイナミクスに基づく時系列データ群は，競合モジュラーネットワーク [11] を用いることによって北海道，日本海側，…，と気象ダイナミクスの類似した地域ごとに分類されることが確認されている [40].

mnSOMはモジュラーネットワークにSOMのWTA，近傍関数を導入したものと考えることもできるので，mnSOMによって日本各地の気象時系列データ群を気象ダイナミクスの類似した地域ごとに分類することは可能だろう．しかも，mnSOMの補間機能により訓練で与えた都市間における中間的な気象ダイナミクスを自己組織的に生成するだろう．すなわちmnSOMは気象ダイナミクスの類似度に基づいた自己組織化マップ，“日本気象マップ”を生成するだろう．

本章では気象データを用い，mnSOMが期待した特製を有し実課題にも応用ができかを計算機シミュレーションによって示す．なお，本章の内容は [46, 45, 44] で報告されている．

5.2 シミュレーションの枠組み

本シミュレーションでは，九州各地20都市（図5.4）の2000年100日分の気圧，気温，湿度，日照時間を要素とした気象データである*1．気象時系列データの例を図5.1に示す．図5.4中のA–Jの10都市を学習クラスとして用い，図5.4中のk–tの10都市をテストクラスとして用いた．

気象データは各都市ごとに確率密度関数$p(x)$が異なるために，全要素に対して平均0に正規化されている．これによって気象要素からバイアス情報を取り除いた気象の変動情報しか残らなくなる．このため，本課題ではmnSOMの1つのモジュールに図5.2のような3層MLPとバイアスを学習するネットワークの2つをハイブリッドしたモジュールを用いる．MLPは一昨日，昨日，今日の気象を入力とし，明日の気象を予測するように学習が行われる．つまり，MLPは各都市のバイアス情報をのぞいた気象の変動値のみを学習する．一方，バイアス部は各都市における各気象要素の100日分の平均値を学習する．つまり，バイアス部は100日を通して暖かかったのか寒かったのか、湿っていたか湿っていなかったかなどの情報を学習する．シミュレーションにおける学習パラメータは表5.1に示す．

5.3 シミュレーション結果

図5.3に結果を示す．各格子がmnSOMの各モジュールに対応しており，格子内のアルファベットは対応するアルファベットの都市（図5.4参照）のBMMを示す．テストクラス（k–t）に関しては学習終了後に，結合荷重を固定したmnSOMにテストクラスを入力させBMMを決定させた．またマップ上でアルファベット同士を結んでいる線は，図5.4に示す線との関連を示す．結果から，気象データを学習させただけにもかかわらず，九州都市の地理的位置のトポロジーを保存したマップが生成された．つまり，mnSOMは“九州地区の気象マップ”の生成に成功したといえる．さらに，学習終了後にBMMの気象データの予測を見たものを図5.1に示す．実線が本来の気象データ，点線がBMMが予測した気象である．結果より，BMMはおおむね予測にも成功していることがわかる．

5.4 考察

シミュレーションによってmnSOMは九州地区気象マップを獲得することができた．

さて，シミュレーションは九州地区で行った．日本全国ではなく九州地区で行った理由

*1 気象データは気象庁年報のものである

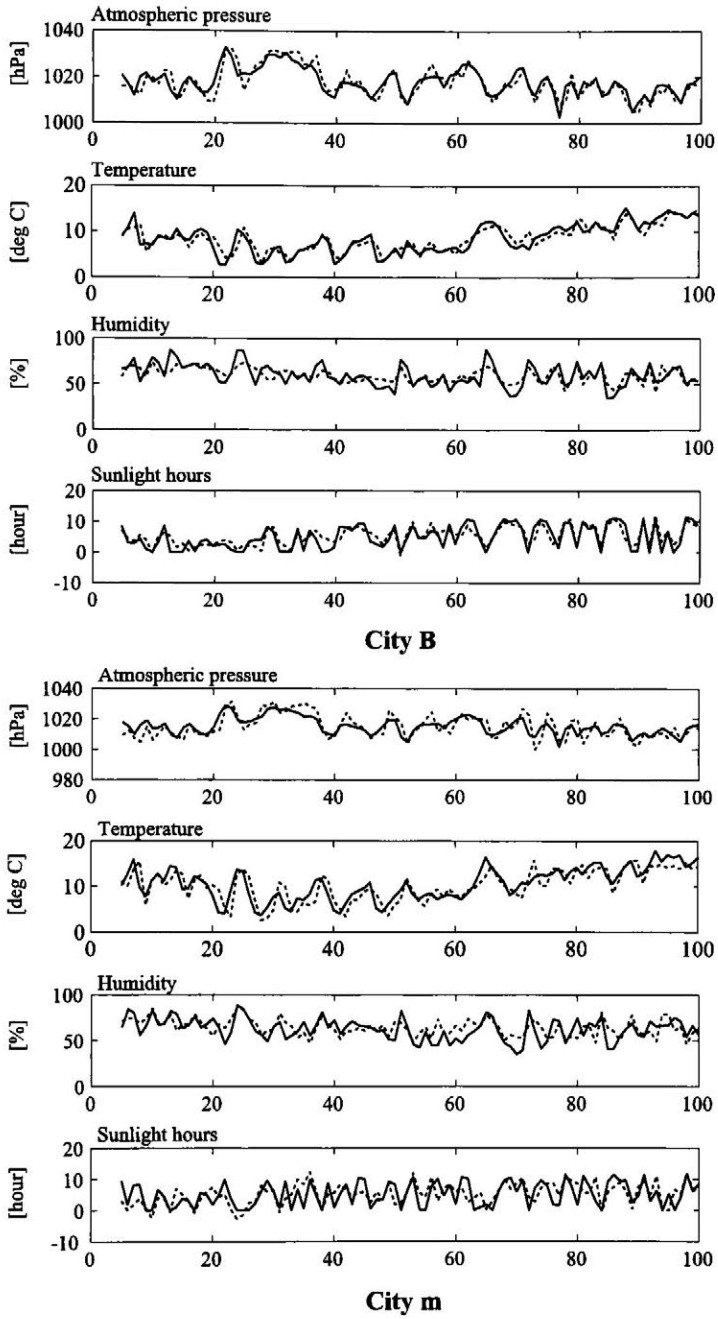


図 5.1 気象データの例（図 5.4 中の B と m の都市）．実線は実際の気象時系列データ，破線は各都市の BMM が実現した気象時系列データ

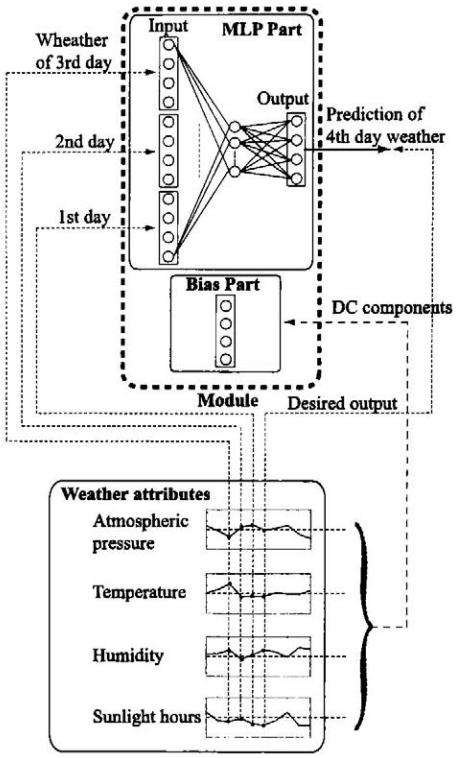


図 5.2 気象時系列データ集合のマップを生成する mnSOM のモジュールのアーキテクチャ

表 5.1 mnSOM による気象ダイナミクスのマップ生成におけるシミュレーション条件

Parameters of MLP module	
Number of units	
Input layer	12
Hidden layer	5
Output layer	4
Bias part	4
Learning constant η	0.01
Parameters of mnSOM	
Map size K	225 (15×15)
Neighborhood function size	
σ_0	15.0
σ_∞	2.0
τ	300
Safety factor β	0.9

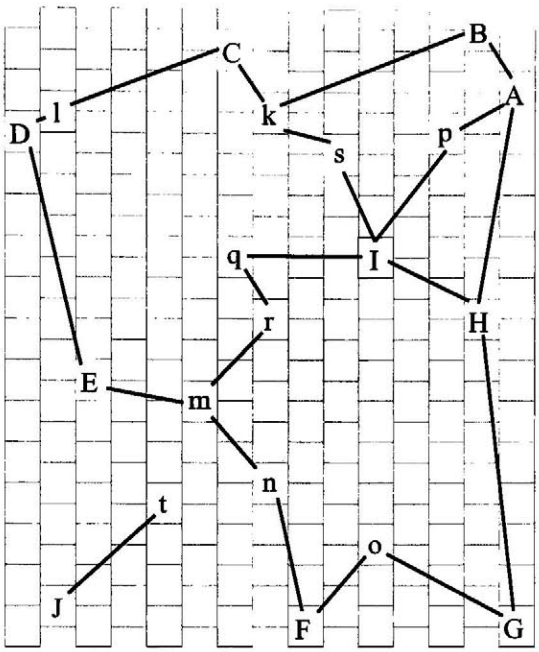


図 5.3 mnSOM による九州地区気象マップ

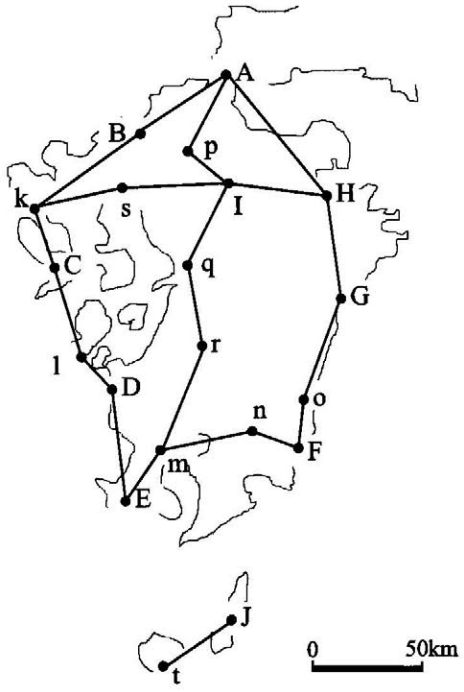


図 5.4 九州地区の地図

は，mnSOMの機能を結果としてわかりやすく見るためである．では日本全国の都市を使用したらどうなるだろうか．日本全国の都市を使用してシミュレーションを行ってみた．使用したデータは日本各地149都市（図5.5(b)）の2000年1月（31日分）の気圧，気温，湿度，日照時間を要素とした気象データである．シミュレーションの枠組み等は九州の場合と同じである．結果を図5.5(a)に示す．各格子内の都市名と色は対応するモジュールが勝者となった都市と地域の色（図5.5(b)）を示す．結果より似た気候の地域がおおむねまとまりさらに，地理的に近い都市もおおむね近い位置にマップされていた．つまり太平洋側，日本海側ともに北海道から九州まで各都市が地理的に連続してマップされていた．

またマップを生成した後，図5.5(a)内の表に示す都市をテスト入力として与えた．その結果を図5.5(a)内のマップ上に示す．アルファベットは対応するモジュールが勝者となった都市を示す．結果よりテストで用いられた各都市は期待される位置にマップされた．

以上より日本全国の都市を使用した場合も地理的トポロジーを保存したマップが生成されたと言っていいだろう．しかし，今後さらにマップをどのように利用するか，気象との関連性などの検証をしていかなければならない．

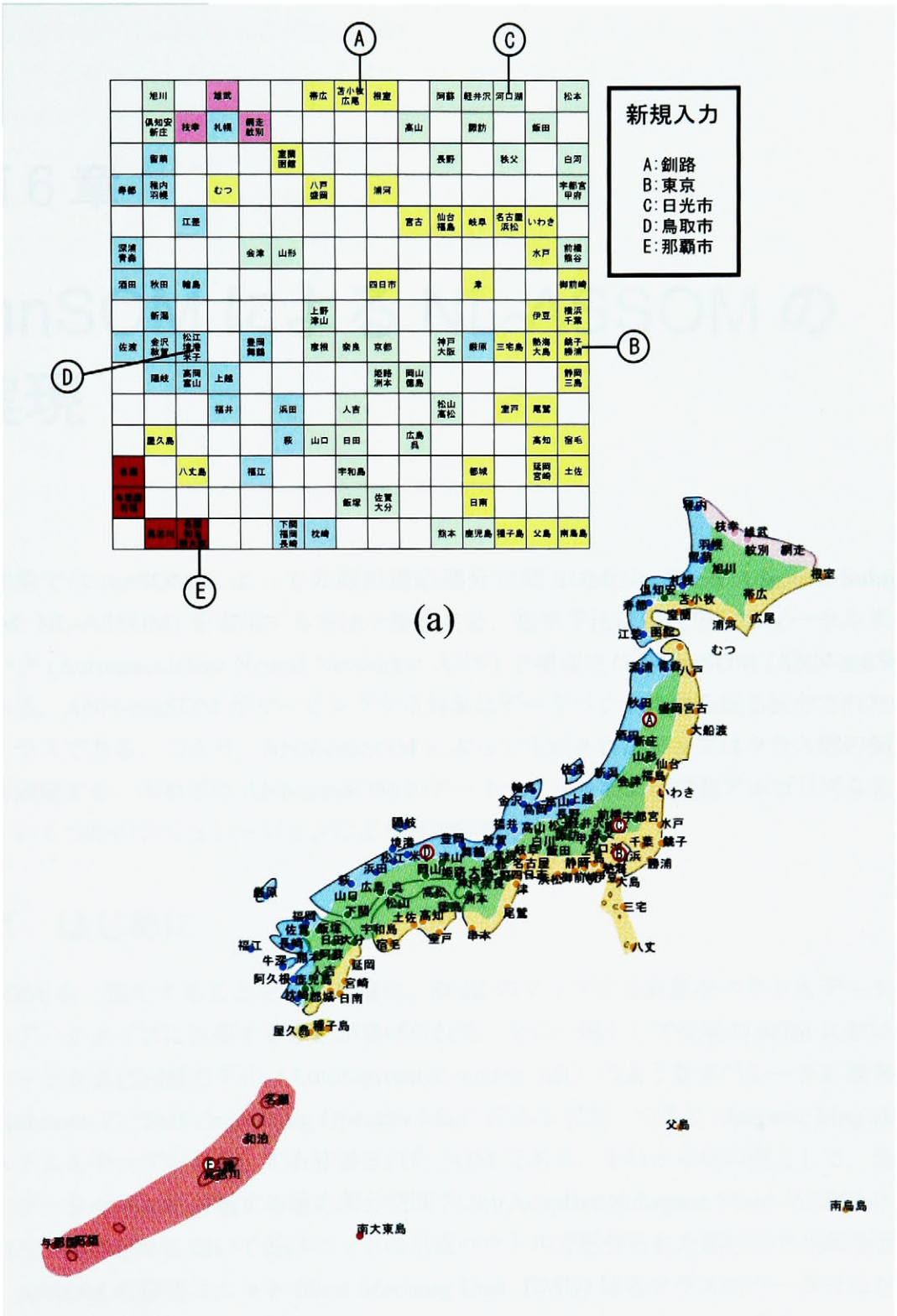


図 5.5 (a)mnSOM による日本気象マップ．(b) 日本地図．太平洋側，日本海側，内陸，オホーツク海側および南西諸島の地域はそれぞれ黄，青，緑，紫，赤で区分している．

第 6 章

mnSOM による NL-ASSOM の実現

本章では mnSOM によって非線形適応部分空間 SOM(Nonlinear Adaptive Subspace SOM: NL-ASSOM) を実現する方法を提案する．提案手法は自己想起ニューラルネットワーク (Autoassociative Neural Networks: ANN) で構成される mnSOM (ANN-mnSOM) である．ANN-mnSOM がマッピングする対象はデータベクトルから成る区分された複数のクラスである．つまり，ANN-mnSOM によって生成されたマップはクラス間の類似関係を表現する．本章では ANN-mnSOM のアーキテクチャおよび学習アルゴリズムを説明し，いくつかのシミュレーションにより有効性を示す．

6.1 はじめに

SOM を一般化することを考えた場合，SOM のマップする対象をベクトルデータから他のデータタイプに拡張することが挙げられる．その一例として従来の SOM における各ユニットを自己回帰モデル（Autoregressive model: AR）のようなオペレータに置き換えた Kohonen の “Self-Organizing Operator Map” がある [23]．つまり Operator Map は線形システムをマップの対象とする拡張された SOM である．それから他の例として，区分されたデータセットを分類する適応部分空間 SOM(Adaptive Subspace SOM: ASSOM) がある [25]．ASSOM において各ユニットは基底ベクトルで形作られた線形部分空間を実現する．ASSOM の勝者ユニット (Best Maching Unit: BMU) は各クラスのデータ分布を最も良く近似するユニットとなる．つまり ASSOM は “参照点” が “参照部分空間” に拡張された mnSOM と最も近い一般化 SOM として見なすことができる．以上の Operator Map と ASSOM の両手法は非線形な場合に自然と拡張できるだろう．しかし非線形課題における適切な SOM の適応アルゴリズムが依然として提案されていない．

これに対して，私はモジュラーネットワークSOM（mnSOM）を提案した．mnSOMは機能モジュール（例えばMulti-Layer Perceptron）を配列させて構成されたネットワークである．これによりmnSOMは非線形関数や非線形ダイナミカルシステムをマップの対象としてあつかえる[44, 10, 9]．このためmnSOMは非線形Operator Mapの解決法であろう．さらにもし機能モジュールとして非線形PCAを実現するニューラルネットワークを用いれば，mnSOMは非線形ASSOM(Nonlinear ASSOM: NL-ASSOM)を実現するだろう．

PCAを実現する手法はいくつかあるが，私は最もポピュラーな手法である自己想起ニューラルネットワーク(Autoassociative Neural Networks:ANN)をmnSOMの機能モジュールとして用いた．ANNはMLPのバリエーションの1つで，入力ベクトルを出力として再帰させるように学習を行う．3層のANN(3L-ANN)は線形部分空間上のデータ分布を近似し，5層のANN(5L-ANN)は湾曲した部分空間（つまり多様体）の分布を実現する．つまり3L-ANNと5L-ANNは線形あるいは非線形PCAに相当する．これより5L-ANNを機能モジュールとするmnSOM(5L-ANN-mnSOM)はNL-ASSOMを実現すると考えられる．

本章ではANN-mnSOMのアーキテクチャ，学習アルゴリズムを説明し，いくつかのシミュレーションにより有効性を示す．なお，本章の内容は[43]で報告されている．

6.2 理論的な枠組み

ANNはMLPのバリエーションの1つで，中間層をはさんで入力側と出力側が鏡像になるようなアーキテクチャを持つ（図6.1）．ANNは教師信号を入力と同一にして学習は行われる．つまりANNの結合荷重は入力ベクトルと対応する出力ベクトルとの平均2乗誤差が小さくなるように更新される．このためANNは入力と同じベクトルを出力側で想起するようになる．このとき入力ベクトルの次元N_Dは中間層ユニットの数N_{BN}よりも少ないため，いかなる入力ベクトル$\boldsymbol{x} \in \mathbb{R}^{N_D}$でも想起するわけにはいかない．想起可能なのは$N_D$次元空間内の$N_{BN}$次元部分空間に所属するデータだけである．したがってANNは与えられたデータベクトルの集合が1つでも多く部分空間内に含まれるように学習が行われる．すなわち学習の結果，ANNの結合荷重はデータベクトルの分布をN_{BN}次元の多様体で近似することになる．3層ANN(3L-ANN)の場合，ANNが近似できる部分空間はN_{BN}次元の超平面である．すなわちデータベクトルをN_{BN}次の主成分分析(PCA)にかける場合と等価である[38]．一方，5層ANN(5L-ANN)が近似する部分空間はN_{BN}次元の超曲面となり，5L-ANNはN_{BN}次の非線形主成分分析(NL-PCA)に相当する[26]．

以上のような特性を持つANNをmnSOMの機能モジュールとした場合，各モジュール

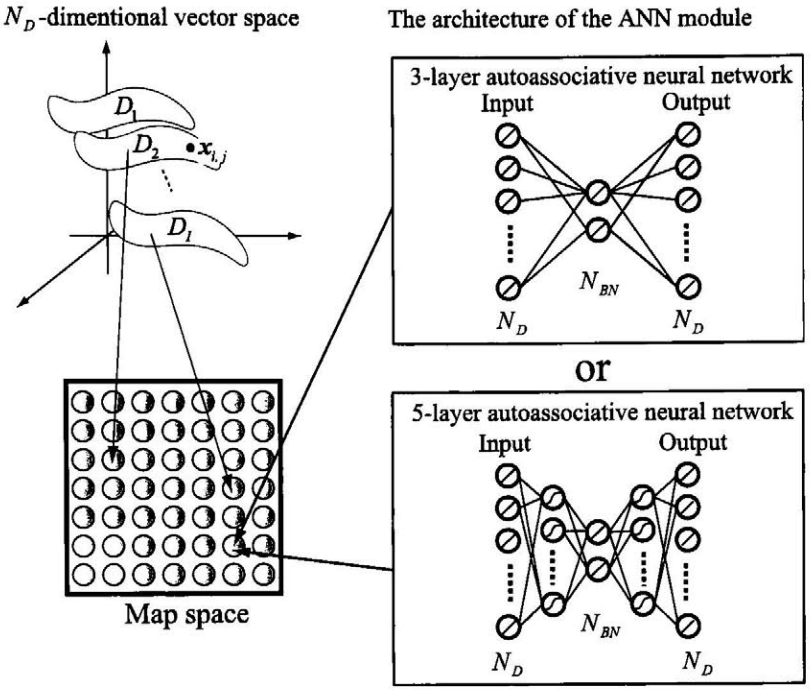


図 6.1 ANN-mnSOM のアーキテクチャ

は「コードベクトル」の代わりに「コードマニホールド」を表現することになる．そして mnSOM 全体では各モジュールが表現する多様体の集合を自己組織化マップとして表現すると期待される．

今，I 個のクラスがあり $(C_1, ..., C_I)$，各クラスからは J 個のサンプルベクトルが観測されているとする．また i-th クラス C_i のサンプルデータ集合 $D_i = \{\boldsymbol{x}_{i1}, ..., \boldsymbol{x}_{iJ}\}$ は確率密度関数 $p_i(\boldsymbol{x}) = p(\boldsymbol{x}|\boldsymbol{\theta}_i)$ から成るとする．ここで θ_i は $p_i(\boldsymbol{x})$ を決める隠れパラメータである．さらに $p_i(\boldsymbol{x})$ は θ_i が変わると連続的に変化すると仮定する．ANN-mnSOM は図 6.1 に示すようなアーキテクチャと枠組みとなっている．ANN-mnSOM において各モジュールは格子状に配置されマップ空間を表現する．つまり ANN モジュールはマップ空間においてポジションを固定されている．このとき ANN-mnSOM は K 個の ANN モジュール $\{M^1, ..., M^K\}$ を有し，各モジュールのマップ空間における座標は $\{\boldsymbol{\xi}^1, ..., \boldsymbol{\xi}^K\}$ とする．また結合荷重 $\boldsymbol{w}^k$ は M^k によって実現された多様体 U^k を表現する．

ANN-mnSOM の目的は ANN モジュールによって確率密度関数の集合 $\{p_i(\boldsymbol{x})\}$ を近似することと $\{C_i\}$ のマップ生成を同時に行うことである．そして生成されたマップには $\{p_i\}$ の類似（または相違）度が反映されると期待できる．具体的に言うと，ANN-mnSOM のタスクは (i) 多様体の集合 $\{U^k\}$ によって $\{p_i(\boldsymbol{x})\}$ を推定し，(ii) 隠れパラメータ空間からマップ空間へ位相同型写像 $\psi : \boldsymbol{\theta} \to \boldsymbol{\xi}$ をすることである．その結果として，確率密度関数の集合 $\{p_i(\boldsymbol{x}|\boldsymbol{\theta}_i)\}$ は $\{p_i(\boldsymbol{x}|\boldsymbol{\xi}_i)\}$ で置き換えるられる．つまり $p(\boldsymbol{x}|\boldsymbol{\theta})$ と $\{\boldsymbol{\theta}\}$ の代替表

現である $(\boldsymbol{x}|\boldsymbol{\xi})$ と $\{\boldsymbol{x}_i\}$ を推定するものと見なすことができる.

ANN-mnSOMの学習アルゴリズムはMLP-mnSOMの場合と同じである. 各クラスに対して勝者モジュール(BMM)を決め, BMMとその周辺のモジュールは他のモジュールより大きな学習率を獲得する. このアルゴリズムは従来のSOMを素直に拡張したものである. しかし, 本章では他のアプローチとして, 拡張したEMアルゴリズムを用い説明をする.

今, 入力ベクトルを $\boldsymbol{x}$, $\hat{\boldsymbol{x}}^k$ を k-th ANNモジュール M^k の出力とする. $\hat{\boldsymbol{x}}^k$ は $\boldsymbol{x}$ を多様体 U^k に写像したものだから, C_i と U^k の間の距離は次式のようにサンプルデータに対する平均誤差によって近似される二乗誤差 $\| \boldsymbol{x} - \hat{\boldsymbol{x}}^k \|^2$ の期待値で定義できる.

$$L^2(C_i, M^k) = \int \| \boldsymbol{x} - \hat{\boldsymbol{x}}^k \|^2 \, p_i(\boldsymbol{x}) d\boldsymbol{x} \tag{6.1}$$

$$\simeq \frac{1}{J} \sum_{j=1}^{J} \| \boldsymbol{x}_{ij} - \hat{\boldsymbol{x}}_{ij}^k \|^2 \equiv E_i^k \tag{6.2}$$

C_i のBMMは最も二乗平均誤差が小さいモジュール M_i^* として定義される. したがって, M_i^* の添え字を k_i^* とするならば k_i^* は次式で与えられる.

$$k_i^* = \arg\min_k E_i^k \tag{6.3}$$

$\boldsymbol{\xi}_i$ の暫定的な推定値はBMM$\boldsymbol{\xi}_i^* = \boldsymbol{\xi}^{k_i^*}$ によって与えられる. つまり, 次式のように $\boldsymbol{\xi}_i^*$ の周囲にガウス分布に従って $\boldsymbol{\xi}_i$ が分布していると仮定する[*1].

$$p(\boldsymbol{\xi}_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(\boldsymbol{\xi}_i - \boldsymbol{\xi}_i^*)^2}{2\sigma^2} \right] \tag{6.4}$$

式6.4の $\boldsymbol{\xi}_i$ は連続変数であるが, モジュールは離散的に配置している. したがって, $\boldsymbol{\xi}_i$ は最も近いモジュールの位置 $\boldsymbol{\xi}^k$ に量子化される. 今, R^k を M^k に所属する領域とするならば, M^k に所属する $\boldsymbol{\xi}_i$ の離散的確率 $P(\boldsymbol{\xi}_i \in R^k) = P(M^k|\boldsymbol{\theta}_i)$ は次式で与えられる.

$$P(M^k|\boldsymbol{\theta}_i) = \int_{\boldsymbol{\xi} \in R^k} p(\boldsymbol{\xi}|\boldsymbol{\theta}_i) d\boldsymbol{\xi} \simeq \frac{S(R^k)}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(\boldsymbol{\xi}_i - \boldsymbol{\xi}_i^*)^2}{2\sigma^2} \right] \tag{6.5}$$

ここで $S(R^k)$ は R^k の大きさを与え, この大きさは全てのモジュールに対して同じであるとする. 式6.5はいわゆる従来のSOMにおける近傍関数である. そして事後確率 $P(\boldsymbol{\theta}_i|M^k)$ はベイズ則によって次式のように与えられる.

$$P(\boldsymbol{\theta}_i|M^k) \equiv \phi_i^k = \frac{\exp\left[-(\boldsymbol{\xi}^k - \boldsymbol{\xi}_i^*)^2 / 2\sigma^2 \right]}{\sum_{i'=1}^{I} \exp\left[-(\boldsymbol{\xi}^k - \boldsymbol{\xi}_{i'}^*)^2 / 2\sigma^2 \right]} \tag{6.6}$$

[*1] $p(\boldsymbol{\xi}_i)$ は $p(\boldsymbol{\xi}_i|\boldsymbol{\xi}_i^*)$ と書くほうがより正確であるが, ここでは条件部を簡素化のため省略する

ここで，k-th モジュールの二乗誤差の期待値 $\langle E^k \rangle$ は次式で評価される．

$$\langle E^k \rangle = E\left[\| \boldsymbol{x} - \hat{\boldsymbol{x}}^k \|^2 \right] \tag{6.7}$$

$$= \sum_{i=1}^{I} \int \| \boldsymbol{x} - \hat{\boldsymbol{x}}^k \|^2 \, p(\boldsymbol{x}|\boldsymbol{\theta}_i) P(\boldsymbol{\theta}_i|M^k) d\boldsymbol{x} \tag{6.8}$$

$$= \sum_{i=1}^{I} \phi_i^k \int \| \boldsymbol{x} - \hat{\boldsymbol{x}}^k \|^2 \, p_i(\boldsymbol{x}) d\boldsymbol{x} \simeq \sum_{i=1}^{I} \phi_i^k E_i^k \tag{6.9}$$

$\langle E^k \rangle$ を評価することは E-step に相当する．その後，最急降下法によって期待値誤差 $\langle E^k \rangle$ を最小にするように各モジュールを学習する．つまり，k-th モジュールの結合荷重 $\boldsymbol{w}^k$ は以下の式によって繰り返し更新される．

$$\Delta \boldsymbol{w}^k = -\eta \frac{\partial \langle E^k \rangle}{\partial \boldsymbol{w}^k} = -\eta \sum_{i=1}^{I} \phi_i^k \frac{\partial E_i^k}{\partial \boldsymbol{w}^k} \tag{6.10}$$

このプロセスは M-step に相当する．式 6.10 は従来のバックプロパゲーションアルゴリムの学習率を各クラスに対して $\eta\phi_i^k$ としたものである．つまり BMMM_i^* とその近傍のモジュールはデータ集合 D_i を他のモジュールよりも学習する．

以上の式は次のようにまとめることができる．式 6.2 と 6.3 は各クラスに対する BMM を決定する *Competitive process* であり，式 6.6 は学習率 $\{\phi_i^k\}$ を計算する *Cooperative process* である．そして最後に *Adaptive process* として式 6.10 によって結合荷重を更新する．この 3 つのプロセスをネットワークの状態が落ち着くまで σ を徐々に減少させながら繰り返す．

6.3 シミュレーション

6.3.1 正弦波集合のマップ

はじめに，3L-mnSOM が ASSOM と等価であることをシミュレーションで示す．このことは Zhang らがすでに，手書きアラビア数字の認識において報告している [50]．ここではとても簡単なタスクとして正弦波ファミリーをマップすることを目的とした．

ある周波数・位相における正弦波の時系列データ集合は超多次元空間の 1 点に対応し，位相が変化すれば時系列データ集合は 2 次元多様体上に分布する．また周波数が変化すると，各周波数ごとに異なる部分空間に 2 次元多様体を形成する．すなわち周波数・位相の異なる正弦波集合族は超多次元空間上で複数の 2 次元多様体を形成し，各多様体は 2 次の線形部分空間で近似できる．本シミュレーションでは，3-ANN-mnSOM でこの複数の 2 次元多様体に対して ASSOM を実現できるか検証した．本シミュレーションでは周波数

表 6.1 3-ANN-mnSOM による正弦波集合のマップ生成におけるシミュレーション条件

$x_{ij}(t)$ の範囲	$[-1.0, +1.0]$
C の周波数 F_C	5.0
訓練クラス	7 音 (C,D,E,$F_\sharp$, $G_\sharp$, $A_\sharp$,C(2))
テストクラス	6 音 ($C_\sharp$, $D_\sharp$, F, G, A, B)
モジュール数	20 個
中間層ユニットの数	2 個

の異なる 13 個（13 クラス）の正弦波を用いた．すなわち次式で生成された正弦波集合を 3-ANN-mnSOM に与えた．

$$s_{ij}(t) = A_{ij} \sin(2\pi f_i t + \phi_{ij}) \quad (i = 0, ..., 12; j = 1, ..., L) \tag{6.11}$$

f_i は周波数であり，$f_i = f_0 \cdot 2^{i/12}$ で与えられる．これは 1 オクターブ離れた音を 13 等分した平均律音階に相当し，$f_0, f_1, f_2, ..., f_{12}$ はそれぞれ $C, C_\sharp, D, ..., C(2)$ に相当する．そして各周波数ごとに位相 θ_{ij}，振幅 A_{ij} をさまざまに変えて得られた L 個の時系列データ集合

$$\boldsymbol{x}_{ij} = \{s_{ij}(t)\} = [s_{ij}(0), s_{ij}(1), ..., s_{ij}(T-1)]^T \tag{6.12}$$

を 3-ANN-mnSOM の入力とした．すなわち 3-ANN-mnSOM の各モジュールは T 次元の入出力を持つ ANN となる．また 2 次元多様体を 2 次の線形部分空間で近似するために ANN の中間層ユニットの数は 2 個とした．本シミュレーションでは $f_1 = 5$, $T = 100$, 時系列データのサンプリング周波数を $f_s = 50 f_0$ とし，サンプルされる正弦波に両側減衰のガウス窓関数をかけてある．また 13 音のうち 7 音 $(C, D, E, F_\sharp, G_\sharp, A_\sharp, C(2))$ を訓練クラスとして用い，残りの 6 音 $(C_\sharp, D_\sharp, F, G, A, B)$ をテストクラスに用いた．学習の結果，3-ANN-mnSOM の各モジュールの中間層ユニットは 2 次の線形部分空間を表す 2 次の基底関数を発現し，さらに各モジュールの基底関数はマップ上で連続的に変化すると期待できる．シミュレーション条件は表 6.1 に示す．

図 6.2 にシミュレーション結果を示す．図中の波形は各モジュールの実現した基底関数を示している．なお，この結果に達したとき，各モジュールは勝者となる音の波形を恒等写像できていた．結果を見ると，隣接するモジュールは似た基底関数形を実現しており，実現された基底関数形は連続して変化している．また訓練終了後に訓練およびテストの音を 3-ANN-mnSOM に与えた．そのときの各音の勝者モジュールを図 6.3 に示す．この結果，個々のモジュールが学習した音の周波数が，連続的に変化していることがわかる．

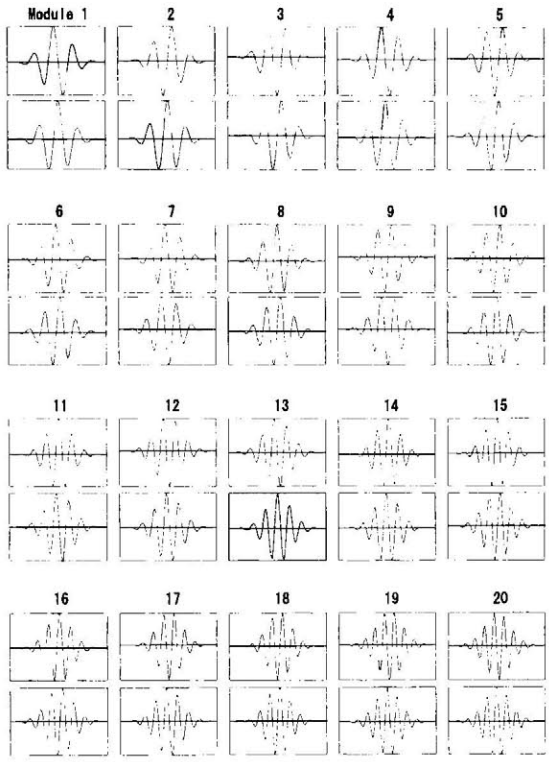


図 6.2 3.1 の結果：各モジュールにおける二つの中間層ユニットが実現した基底関数を示す.

C	C#	D	D#	E	F	F#	G	G#	A	A#	B	C(2)
0	1	1	3	5	7	8	10	12	14	15	17	19

図 6.3 各音の勝者モジュール番号

これより 3-ANN-mnSOM の隣接するモジュールは類似した基底関数を実現し，ネットワーク全体では基底関数を基にした自己組織化マップができたと考えられる．すなわち，3-ANN-mnSOM は線形の適応部分空間 SOM(ASSOM) を実現できた．また学習に用いなかったテストの波形に対しても，期待した場所が勝者となった．このことは，中間的な基底関数を mnSOM が補間したものと考えられる.

6.3.2 周期振動波形の自己組織化マップ

次に 3.1 のシミュレーションと同じ枠組みで，波形を正弦波だけでなく三角波，矩形波も用いたシミュレーションを行った．この場合もデータベクトルは 2 次元多様体上に分布するが，しかし線形部分空間は作らない．このため本シミュレーションでは非線形部分空間 SOM を実現するために 5-ANN-MLP を用いる．本シミュレーションでも，各波形ごとに周波数比の異なる 13 個の音を用いる．また $f_0 = 2$ とし，サンプリング周波数

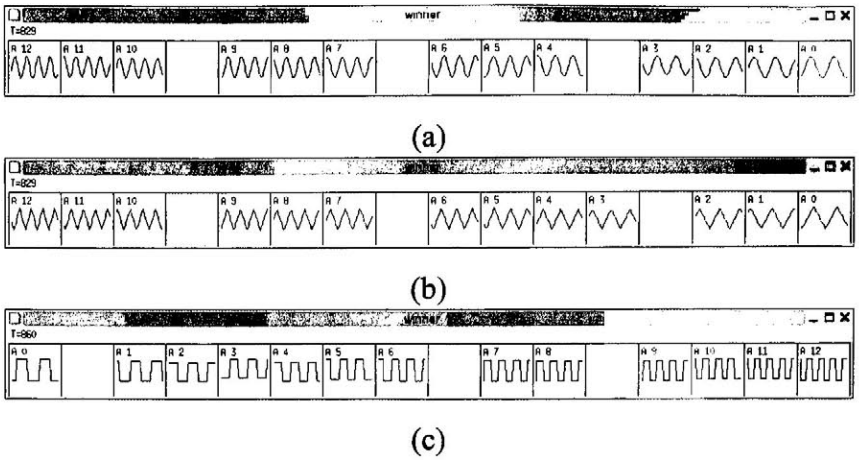


図 6.4 3.2 の結果 (ケース (1)) : (a) 正弦波 (b) 三角波 (c) 矩形波

は $f_s = 32f_0$ とした．シミュレーションは次の 2 ケースについて行った．(1) 各波形ごとに mnSOM のマップを作成する (16 個のモジュールは 1 次元に並べる)．(2) 全ての波形を mnSOM に与えマップを作成する (13×13 に 2 次元にモジュールを配置する)．

まず (1) の結果を図 6.4 に示す．波形の描かれている格子は，mnSOM に与えた 13 個の周波数に対して勝者となったモジュールに対応する．結果を見ると，どの波形の場合も周波数の順番でマップが生成された．また正弦波以外の場合は超多次元空間において非線形部分空間を形成するが，その場合でも問題なくマップが形成されることが確認できた．次に (2) の結果を図 6.5 に示す．各々の波形は周波数順に mnSOM のマップ空間を馬蹄形に 1 周して並び，全体としては同心円状のマップが出来上がった．またこの同心円は外側から矩形波，正弦波，三角波の順に並び，波形についても連続的なマップとなった．このように，2 次元マップ上において，クラスが馬蹄形状に配置される理由として，各クラスの部分空間が長細く分布していることが挙げられる．つまり，波形が異なっても周波数が同じ場合，各クラスの分布距離は近いが，一方，同じ波形でも周波数が異なると各クラスの分布距離は遠くなる．これはベクトル空間上に長細く分布しているデータに対して SOM でマップを生成した場合も，マップ上では入力データ空間が曲がってしまう．ゆえに，mnSOM においても，SOM のマップの性質上，周期振動波形は馬蹄形に配置される．このことは 7 章の動的システム集合のマップ結果においても同じことが言える．以上の結果より，5-ANN-mnSOM は NL-ASSOM を実現できたと考えられる．

6.3.3 3 次元物体集合の自己組織化マップ

3 次元物体を様々な角度から撮影してできた写真の集合，すなわち 2 次元射影像群は超多次元ベクトル空間において 2 次元多様体を作る．したがって複数の 3 次元物体が存在す

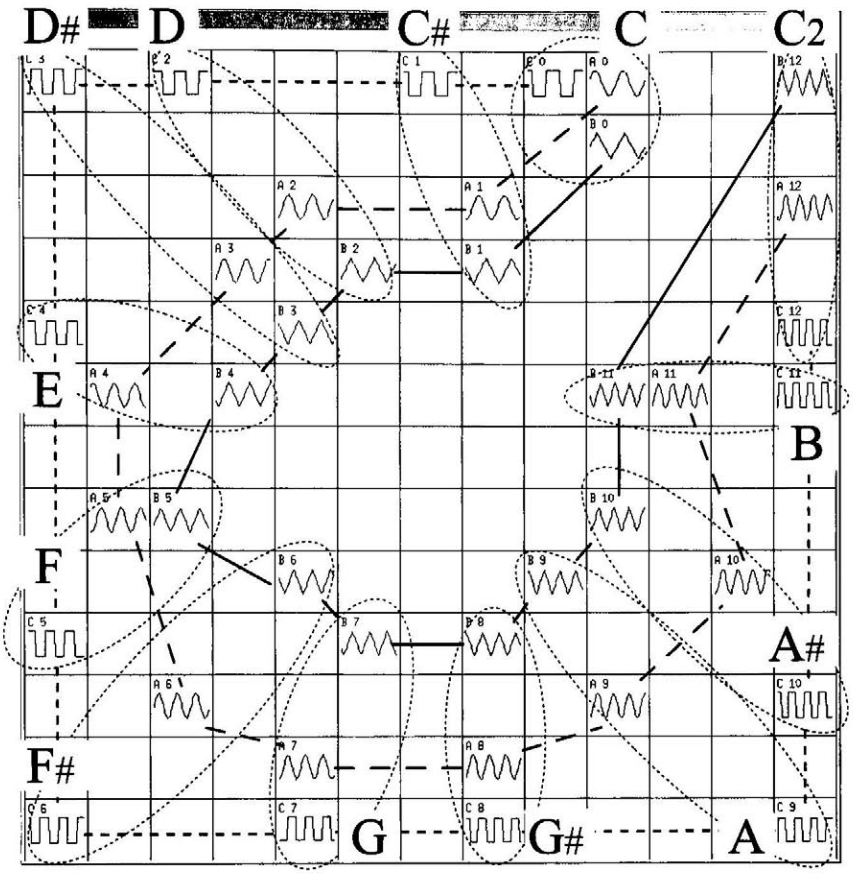


図 6.5 3.2 の結果 (ケース (2)) : 点線で囲まれた部分が同じ音 (周波数) を示す. また同じ波形は同種の線で結んである.

る場合, それらの 2 次元射影像は物体と同じ数の多様体集合を作る. 本シミュレーションでは, これら多様体集合の自己組織化マップを 5-ANN-mnSOM で生成することを目的とする. すなわち 2 次元射影像だけを 5-ANN-mnSOM に与えて学習させることにより, 5-ANN-mnSOM は 3 次元物体の形状を基にした自己組織化マップを生成し, さらにネットワークに 2 次元射影像から 3 次元形状の復元方法を教えなくても, 5-ANN-mnSOM は 2 次元射影像から 3 次元物体を識別すると期待できる. 以上が可能かどうかを確認するために, 人工的な 3 次元物体を用いシミュレーションを行った.

本シミュレーションでは, 3 次元空間に $9 \times 9 = 81$ 点の座標集合から成るグリッド状の 3 次元物体を用いた. この 3 次元物体を 2 次元平面に射影したときの各点の (x, y) 座標が 5-ANN-mnSOM の入力となる. すなわち 81 点 $\times 2 = 162$ 次元ベクトルが 1 個の 2 次元射影像に相当する. 3 次元物体同士の類似度を定量的に評価できるようにするために, 3 次元物体の形状は次式で与えられる. ここで, i-th 物体の第 n 点の 3 次元座標を

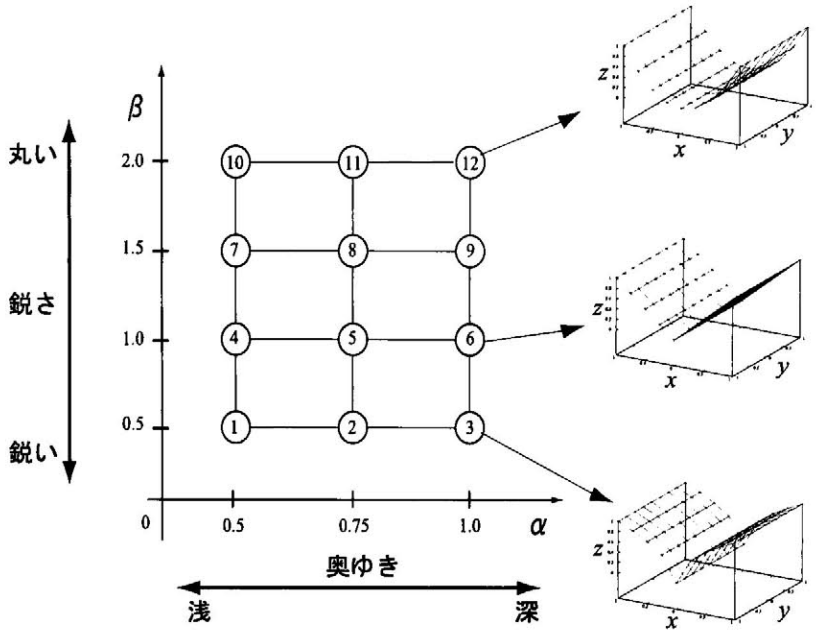


図 6.6 パラメータ空間および人工 3 次元物体の代表例

(x_{in}, y_{in}, z_{in}) とする.

$$z_{in} = \alpha_i |x_{in}|^{\beta_i} \tag{6.13}$$

(6.14)

上式において，α_i，β_i は物体の形状を決めるパラメータであり，α_i は物体の奥行きの深さを，β_i は突起の鋭さを決める．これらを変えることによって 3 次元物体の形状は変化する（図 6.6）．本シミュレーションでは，図 6.6 の (α_i, β_i) パラメータ空間上における 12 個の 3 次元物体を訓練クラスとして使用する．この 3 次元物体を x，y 軸周りに θ，ϕ $(-45° \leq \theta \leq 45°,\ -45° \leq \phi \leq 45°)$ を 5° 刻みで回転させ 2 次元射影像を生成する．つまり，1 つの 3 次元物体につき，361 方向からの 2 次元射影が得られる．このようにしてできた 2 次元射影像だけが 5-ANN-mnSOM に入力されるが，3 次元物体に復元する方法は一切教えない．5-ANN-mnSOM の各層のユニット数はそれぞれ $162, 20, 2, 20, 162$ とした．また，100 個のモジュールを 10×10 の 2 次元に配置した 5-ANN-mnSOM を用いた．他の詳細は表 6.2 に示す．

図 6.7 に結果を示す．各格子内に描かれた図形は，対応するモジュールが獲得した 3 次元物体を 1 方向から見た形状である．また，太枠で示したモジュールは学習に用いた 12 個の 3 次元物体に対する勝者である．結果を見ると，隣り合うモジュールは似た 3 次元物体の形状を獲得していた．また訓練クラスのパラメータ空間におけるトポロジーは，マップ上でも保存されていた．さらに学習させた各 3 次元物体間の中間的 3 次元物体を各モ

表 6.2 mnSOM による 3 次元物体集合のマップ生成におけるシミュレーション条件

ANN の各層の構成	162-20-2-20-162
モジュール数	$10 \times 10 = 100$
BP の学習率 η	0.01
σ_0	8.0
σ_∞	2.0
τ	40.0

ジュールは実現していた．なお，結果図 6.7 の格子内には 3 次元物体の 1 方向の形状しか示してないが，実際にはどのモジュールも様々な方向の形状を実現していた．つまり，各モジュールは 3 次元物体を識別できていると考えられる．

6.4 考察

mnSOM の基本モジュールに ANN を用いた場合，mnSOM が多様体集合の NL-ASSOM を実現するかどうかを，多様体集合を用いたいくつかのシミュレーションによって検証した．その結果，mnSOM は学習対象の多様体集合に対する NL-ASSOM を獲得した．これは mnSOM の競合作用により，与えられた各多様体に勝者モジュールが割り当てられ，各多様体ごとに各勝者モジュールが適応部分空間を形成したためである．さらに近傍関数の作用により，勝者モジュール以外のモジュールは，与えられた多様体集合の各適応部分空間を補間するような部分空間を実現し，それはマップ上で連続的に配置された．

また多様体を表現するニューラルネットワークの代表例として SOM 自身が挙げられる．ベクトルユニットを 2 次元に配置した SOM は，高次元データベクトル空間の 2 次元多様体を表現する．多様体が n 次元であれば SOM のマップも n 次元にすればよい．このとき，与えられたデータに対する勝者ユニットの位置が多様体上におけるデータの座標を表す．したがって mnSOM の基本モジュールとして SOM を採用しても多様体集合のマップを形成することができる．これについての詳細は以下の論文に書かれている [7, 8, 35].

本論文によって，mnSOM が複雑な多様体集合に対しても自己組織化マップを適用できることが確認された．これにより mnSOM は多くの実用課題で応用可能だろう．

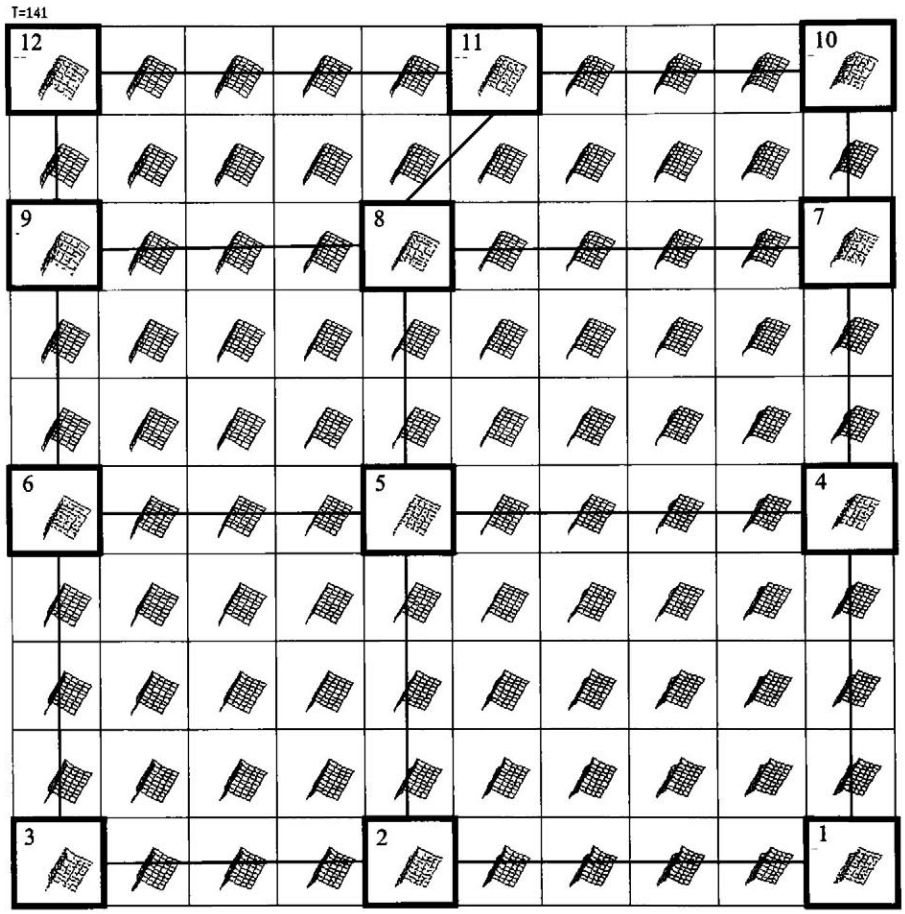


図 6.7 mnSOM による 3 次元物体集合のマップ：訓練で用いた 3 次元物体の勝者を太枠で囲ってある．パラメータ空間との対応を見るために，勝者モジュール間に線を引いてある．

第 7 章

mnSOM による動的システム集合の自己組織化マップ

本章では mnSOM を用いた動的システム集合の自己組織化マップ生成を試みる．すなわち mnSOM の各機能モジュールは 1 個の動的システムの入出力関係を学習し，mnSOM 全体ではシステム同士の類似関係をもとにした自己組織化マップを生成することを期待する．もしこれが可能ならば，非線形制御，ロボットなどの課題に mnSOM を適用できる．つまり mnSOM の各モジュールがそれぞれ異なるパラメータ状態を学習することができ，さらに未学習の中間的な状態も自動生成して，パラメータ空間に相当する自己組織化マップを作ることができるからである．本章では，動的システム集合を扱うための mnSOM のアーキテクチャ，アルゴリズムを説明する．さらにロボティクス応用への準備を視野に入れながら簡単な力学系（動的システム）として減衰振動系の線形システム集合を用いてシミュレーションを行ったのでそれぞれ報告する．なお，本章の内容は [46] で報告されている．

7.1 動的システム集合を扱うための mnSOM のアーキテクチャ，アルゴリズム

7.1.1 アーキテクチャ

図 7.1 は mnSOM のアーキテクチャである．mnSOM の機能モジュールは格子状に配置され，この格子がマップ空間を示す．すなわち各モジュールはマップ上の各点に対応する．mnSOM の各モジュールはそれぞれ 1 個のシステムを学習することができ，モジュールの構造は学習対象に応じて変えることができる．本章では時々刻々と入出力が変化する動的システム集合を取り扱うため，機能モジュールとして Elman のリカレントニューラ

ルネットワーク (RNN) [5] を採用する．すなわち，1 つのモジュールは観測されている時系列データを学習することによって 1 つの動的システムの入出力特性を獲得できる．そして mnSOM 全体では各システムにおける入出力特性の類似度に基づいて自己組織化マップを生成すると期待できる．

今，I 個の動的システム $(C_1, ..., C_I)$ から，それぞれ入出力時系列集合 $\{\boldsymbol{x}_i(t), \boldsymbol{y}_i(t)\}$ が観測されたとする．また，各システムの入出力特性 $f_i(\cdot)$ は未知とする．すなわち，$\boldsymbol{y}_i(t) = f_i(\boldsymbol{x}_i(t); \boldsymbol{\theta}_i)$ が成り立つ．ここで，$\boldsymbol{\theta}_i$ は $f_i(\cdot)$ を決めるシステムのパラメータである．このとき mnSOM は以下のタスクを満たすことを目的とする．

(1) 観測された $\{\boldsymbol{x}_i(t), \boldsymbol{y}_i(t)\}$ から $f_i(\cdot; \boldsymbol{\theta}_i)$ を同定する．
(2) $\{\boldsymbol{x}_i(t)\}$ をパラメータ θ_i の順番で並び替え，自己組織化マップ上へ配置する．すなわちパラメータ空間の移送が保存されるようにマッピングする．
(3) 観測波形の存在しない未知のパラメータを持つシステムについても，自己組織化マップ上で内挿補間をし $f(\cdot)$ を推定する．

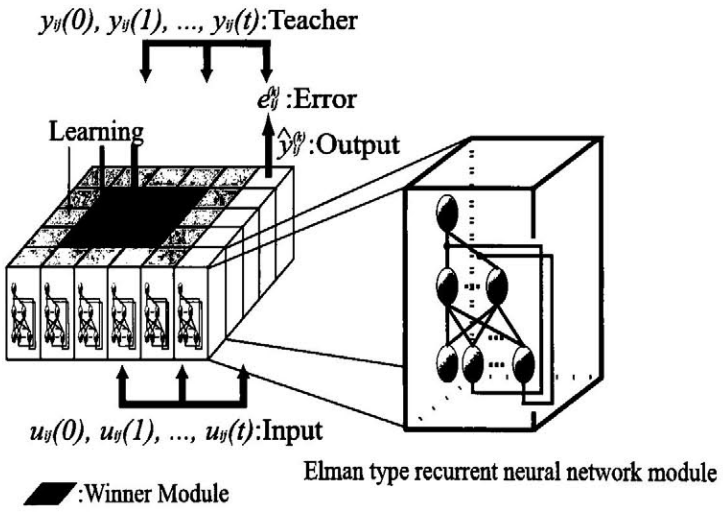


図 7.1 動的システム集合を扱う mnSOM のアーキテクチャ

7.1.2 アルゴリズム

本章における mnSOM のアルゴリズムは通常の mnSOM のアルゴリズムと変わらず，評価，競合，協調，学習プロセスの 4 つのプロセスから成る．

Evaluative process

Evaluative process では各モジュールの出力を計算し，誤差を評価する．今，$\boldsymbol{x}_i(t)$ をすべてのモジュールに入力したとすると，各モジュールからは $\hat{\boldsymbol{y}}_i(t)$ が出力される．$\boldsymbol{x}_i(t)$ に対する k-th モジュールの出力を $\hat{\boldsymbol{y}}_i^k(t)$ とするならば，各モジュールの教師信号 $\boldsymbol{y}_i(t)$ との出力誤差は次式で表される．

$$e_i^k(t) = \frac{1}{2}\|\boldsymbol{y}_i(t) - \hat{\boldsymbol{y}}_i^k(t)\|^2 \tag{7.1}$$

したがって，時系列全体に対する誤差 E_i^k は

$$E_i^k = \sum_{t=0}^{T-1} e_i^k(t) \tag{7.2}$$

となる．これをすべての時系列に対して計算する．

Competitive process

Competitive process では各時系列データに対する勝者を決定する．時系列 $\boldsymbol{x}_i(t)$ に対して誤差 E_i^k が最も小さいモジュールが勝者 k_i^* に選ばれる．つまり

$$k_i^* = \arg\min_k E_i^k \tag{7.3}$$

であり，$\boldsymbol{x}_i(t)$ のダイナミクスを最も良く学習したモジュールが勝者となる．

Cooperative process

Cooperative process では，各モジュールの学習分配率が計算される．学習分配率は次式で定義されるように勝者とその周辺モジュールに多く割り当てられる．

$$\psi_i^k = \frac{\phi(k, k_i^*)}{\sum_j \phi(k, k_i^*)} \tag{7.4}$$

ここで $\phi(k, k_i^*)$ は近傍関数であり．通常はガウス関数を用いて次式のように表す．

$$\phi(k, k_i^*) = \exp\left[-\frac{d(k, k_i^*)^2}{2\sigma(n)^2}\right] \tag{7.5}$$

ただし $d(k, k_i^*)$ はマップ上において，勝者と他モジュール間の距離を意味する．また $\sigma(n)$ は近傍の広さを決める変数であり．学習回数 n に対して単調に減少する．学習分配率 ψ_i^k は，各モジュールがそれぞれの時系列をどれくらいの比率で学習するかを決めるものであり，どのモジュールも総学習量は 1 になるように規格化されている．

Adaptive process

Adaptive process では，学習分配率に応じてモジュールの結合ベクトル $\boldsymbol{w}^k$ の学習が行われる．学習は最急降下法を用いて次式のように行う．

$$\delta \boldsymbol{w}^k = -\eta \sum_{i=1}^{I} \psi_i^k \frac{\partial E_i^k}{\partial \boldsymbol{w}^k} \tag{7.6}$$

Adaptive process の後は再び *Evaluative process* に戻り，ネットワーク全体が収束するまで繰り返す．以上のアルゴリズムにより，ある時系列のダイナミクスを学習したモジュールはますますその時系列に特化するように学習が進む．その際，近傍のモジュールに対しても学習が行われ，その結果としてダイナミクスの類似度を反映したマップが生成される．

7.2 計算機シミュレーション

mnSOMが動的システムの課題に適用できるかをシミュレーションによって確かめた．今回は動的システムの例として図7.2(a)に示すような減衰振動系を用いた．系の運動方程式は以下の式で表される．

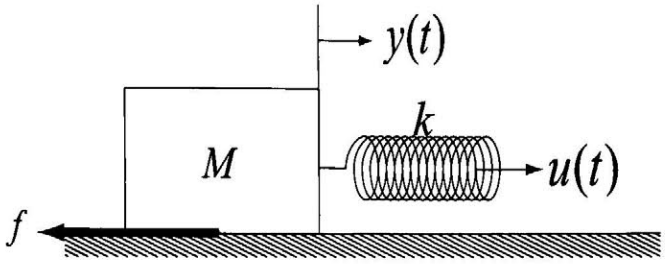
$$M\ddot{y}(t) + F\dot{y}(t) + Ky(t) = x(t) \tag{7.7}$$

この式ではばね定数 K，質量 M，粘性摩擦係数 F の3つのパラメータがあるが，今回は F, K を変えて13種のシステムを用意した（図7.2(b)）．なおシステム1〜9は学習用，システム A〜D はテスト用である．i-th システムへの入力 $x_i(t)$ はホワイトノイズとし，おもりの位置 (つまり出力) を $y_i(t)$ とした．mnSOMには $x_i(t)$ が入力され，教師信号として $y_i(t)$ が与えられる．この結果，図7.2(b)に示すようなシステム間のトポロジーがマップ上で保存されていることが望ましい．その他の実験条件は表7.1に示す．

実験結果

mnSOMが9種の時系列データを学習した結果，生成されたマップを図7.3(a)に示す．各マス目はmnSOMのモジュールに対応しており，マス目内には対応するモジュールが獲得した入出力特性（インパルス応答関数）を示す．また黒のマス目は学習用システムに対して勝者となったモジュールを，灰色のマス目はテスト用システムに対して勝者となったモジュールを示している．

結果を見ると，隣接するモジュールは互いに似たインパルス応答関数を獲得しており，マップ全体を見るとそれが連続的に変化していた．また学習終了後にテスト用時系列データを与えると，パラメータ空間より予想される位置のモジュールが勝者となった．さらに



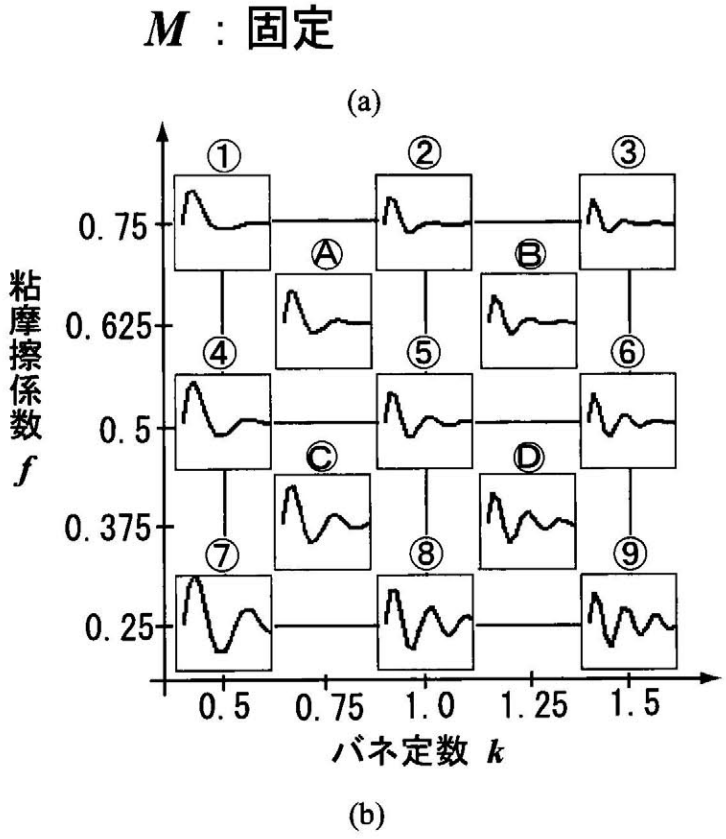


図 7.2 (a) 減衰振動系 (b) パラメータ空間における学習用，テスト用システムのインパルス応答

図 7.2(b) のパラメータ空間におけるトポロジーは，マップ上では図 7.3(b) のようになる．この結果を見ると，トポロジーが保存されてマップされていることがわかる．同様の試行を何度か繰り返したが，この結果は常に一貫していた．

さらに mnSOM は与えられた 9 個のシステムを学習しただけでなく，残り 91 個のモジュールを使って中間的なシステムも自動生成し，さらにそれらを連続的なマップの上に配置することができた．パラメータには無数の組み合わせが存在するが，mnSOM はそのうちの有限少数個のシステムを学習させることで，その他の中間的なシステムを自動生成し内挿補間することができた．

表 7.1 mnSOMによる動的システム集合のマップ生成のシミュレーション条件

M：質量	1.0
F：粘性摩擦係数	（学習用）0.25, 0.5, 0.75
	（テスト用）0.375, 0.625
K：バネ定数	（学習用）0.5, 1.0, 1.5
	（テスト用）0.75, 1.25
外力 $u(t)$ の範囲	$[-0.5, +0.5]$
学習用システムの種類	9種
テスト用システムの種類	4種
モジュール数	100個
中間素子数	5個

7.3 考察

本シミュレーションでは動的システムの例として減衰振動系のダイナミクスをmnSOMに学習させた．mnSOMは与えられた少数個のシステムの特性を学習しただけでなく，システム間を補間するような中間的な特性を自己発現し，連続的なマップを実現した．このとき，注目すべきところは，mnSOMは波形の類似性ではなくシステム特性の類似性に基づいてマップを生成していることである．すなわち，観測された出力波形のサンプリングベクトルを通常のSOMに与えてもSOMは波形の類似性でマップを作るが，mnSOMは観測された入出力時系列からシステムの特性を学習によって同定し，さらにシステム特性の類似度に基づいて自己組織化マップを生成する．これは通常のSOMが持っていない能力である．このため，mnSOMは時系列データを用いる応用課題，例えばロボティクス，制御などの分野では非常に強力なツールとなるだろう．

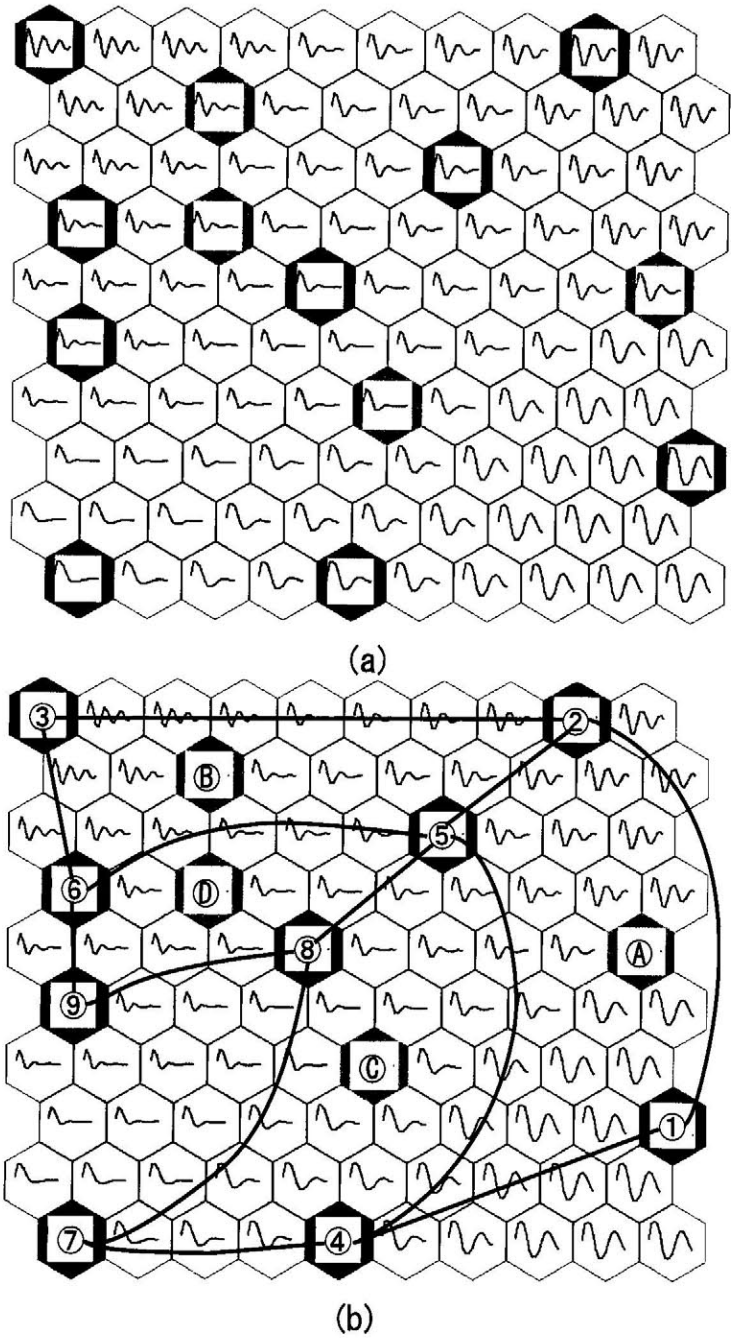


図 7.3 mnSOM による動的システム集合のマップ

第 8 章

考察

8.1 通常型 SOM との比較

mnSOM と通常型 SOM との違いについて述べる．mnSOM は関数空間での近傍関係を基に自己組織化マップを作成する．ベクトル空間で自己組織化マップを生成する通常の SOM でも関数を等間隔でサンプリングし，関数の概形をベクトルとして与えた場合も同様の結果が得られるものと考えられる．しかし mnSOM は，

1. ランダムにサンプルされたデータ集合でも対応できる．
2. 関数形が未知，あるいは入力次元の高い場合でも学習できる．
3. 新規入力に対する出力を各モジュールの出力から推測できる．

といった特徴を持つ．

ここで Kohonen は通常の SOM においてベクトルを扱う静的なユニットを動的なオペレータに置き換えることにより，時間的に変化するデータに対しても SOM を適用できることを示している [23] この提案法を Kohonen は Self-Organizing Operator SOM と呼び，1 つのモジュールに線形な 2 層のネットワークを用いている．また Adaptive Subspace SOM（ASSOM） [25] も同様に通常の SOM におけるユニットを線形素子を用いた 2 層のネットワークに置き換えたモジュラーネットワークであり，各モジュールは線形基底を実現する．これは mnSOM の 1 つのモジュールに 3 層砂時計型パーセプトロンを用いたネットワークと等価である．これらは本手法と類似しているが，mnSOM はそれらの手法を，非線形素子を用いた 3 層のネットワークに拡張したものと考えられる．また本論文では mnSOM の基本モジュールとしてリカレントニューラルネットワークや SOM とハイブリッドしたネットワークを用いている．このように課題にあわせてモジュールを変えることにより，良好な実験結果を得ることができた．以上のような点から，Kohonen の Self-Organizing Operator SOM や ASSOM の考えを発展させた mnSOM は，さらに様々

な実用的課題に広く適用できると考えられる.

8.2 他のモジュラーネットワークとの関連

Jacobs, Jordan らの"mixture of experts" は代表的なモジュール競合学習である [15, 17]. この手法は入力空間を分割し各部分空間毎に各エキスパートネットワークを割り当てる. そして各エキスパートネットワークは各部分空間の入出力関係を学習により実現することができる. このとき各部分空間に対してのエキスパートネットワークの割り当ておよび学習はゲーティングネットワークによって競合的に行われる. このようなモジュール競合学習は様々な手法が提案され各分野で応用されている [48, 42, 41] また古川の提案した期待値動作による競合モジュラーネットはゲーティングネットワークを用いずに, 各モジュールの出力誤差を基にして求められたモジュールの選択確率を用い, 期待値としてネットワークを動作させている [11]. このネットワークは与えられた入力データ集合を局所的なサブモデルの組み合わせで表現し, 入力空間における最適なモデル化と最適なクラス分類を同時に行う. 一方, mnSOM は Jacobs らの競合モジュラーネットのように, ゲーティングネットワークを用いてサブモデルの選択をするものではない. mnSOM は出力誤差が最小となるモジュールを勝者とし勝者および近傍モジュールの学習量を近傍関数によって決定している. ゲーティングネットワークを用いずにモジュールの出力誤差を基にしている点では古川の期待値動作による競合モジュラーネットと同じである. ゆえに, 古川の期待値動作による競合モジュラーネットに SOM における近傍関数を導入したネットワークとも考えられる. またモジュール競合学習では, 入力空間において平均・分散が違うデータ集合が複数存在する場合にその入力空間を分割し, 各モジュールは分割された各入力空間内における入出力関係をそれぞれ学習する. これに対して mnSOM は, 各関数 (システム) の入出力空間毎に各モジュールが割り当てられ入出力関係を学習する.

8.3 mnSOM が扱う課題の問題設定について

本研究において MLP-mnSOM が扱った問題設定をまとめると,

- I 個のデータセット (クラス) があり. 各クラスにおいて J 個のサンプリングデータが観測されている.
- 各クラスの確率密度関数 $p(x)$ は一定とする.
- システムの関数は未知.
- どのデータがどのシステムから観測されたかはわかっている.

となる．気象，周期振動波形，3D 物体の課題では $p(x)$ がクラスごとに異なっていることを除くと，他は同じ設定である．$p(x)$ については後述する．

この他にも mnSOM が直面する問題設定としていくつか考えられる．

まず，数は不明だが多数のクラスがあり，その中からランダムに選択されたクラスから数点のサンプリングデータが観測されるという問題設定が考えられる．この問題設定の場合，データセットが次々と観測され，しかも一度観測されたデータセットは二度と観測されることはない．このため，メモリにデータセットを格納し一括して学習させる一括学習は扱えない．しかし mnSOM はナイーブ，ベイズ SOM のどちらのアルゴリズムでも逐次的な学習をすることでマップを生成することができる．なおこの問題設定ではどのクラスからデータセットが得られたか不明なため，生成されたマップに対してラベル付けをすることはできない．通常の SOM でもこの問題設定ではラベル付けはできない．

次に，数が不明の多数のクラスがあり，クラスの特性が連続的に変化するように選択されたクラスから数点のサンプリングデータが観測されるという問題設定を考える．この問題設定では，観測されたデータセットの元となるクラスの特性が時々刻々と連続的に変化する．例えば，移動するロボットから観測される外界情報のように，環境の変化によってクラスの入出力関係も連続的に変化しながらデータセットが観測される場合がこの問題設定に当てはまる．このような問題設定の場合も，クラス数が未知なので全てのクラスに対して一括学習をすることはできない．しかし，観測された数セットのデータセットに対してベイズ SOM のアルゴリズムを適用しマップを生成することはできる．このとき，1 セットのデータセットに対して学習を行えば通常の逐次学習になる．なおナイーブ SOM のアルゴリズムを使用するならば，学習のスケジューリングが問題となってくる（2.1.1 節，ナイーブ SOM を参照）．

8.4 確率密度関数のクラス依存性

MLP-mnSOM では次式のように関数空間における距離をもとにマップを生成する．

$$L^2(f_i, g^k) = \int \|f_i(\boldsymbol{x}) - g^k(\boldsymbol{x})\|^2 p_i(x) d\boldsymbol{x} \tag{8.1}$$

このとき，理論的に関数同士の距離を測るならば確率密度関数 $p(x)$ がかかわってくる．実際の mnSOM における計算過程でこの $p(x)$ を求めたりすることはないが，観測データセットの $p(x)$ が異なるとさまざまな問題が生じる．以下に $p(x)$ が mnSOM の学習にどのように関わってくるかを説明する．

確率密度関数がクラスに依存せず一定の場合

$p(x)$ がクラスに依存せず一定の場合，モジュールの実現する関数は次式に示すように関数の線形内分点で記述できる．

$$g^k(x) \rightarrow \sum_i \psi_i^k f_i(x) = \psi_1 f_1(x) + \psi_2 f_2(x) + ... + \psi_I f_I(x) \tag{8.2}$$

確率密度関数がクラス間で異なる場合

各クラスにおいてサンプリングデータに偏りがありクラスごとに $p(x)$ は異なる場合，モジュールは次式に示す関数を実現する．

$$g^k(x) \rightarrow \frac{\psi_1^k p_1(x)}{\sum_{i'} \psi_{i'}^k p_{i'}} + \frac{\psi_2^k p_2(x)}{\sum_{i'} \psi_{i'}^k p_{i'}} + ... + \frac{\psi_I^k p_I(x)}{\sum_{i'} \psi_{i'}^k p_{i'}} \tag{8.3}$$

もし，$p(x)$ は異なってもデータ x の分布領域がクラスごとに重なりがあるならば，本来実現して欲しい関数に比べて多少歪みを生じるが計算ができないわけではない．

一方，データ x の分布領域も重ならない場合は，1 つのモジュールが複数のクラスの関数を実現してしまう．この場合，関数同士の距離の計算ができず，mnSOM は関数の類似度を見てマップを作れないこのようなデータセットの場合，各クラス平均を 0 に合わせたり，平均 0，分散 1 に正規化することでマップ生成の不都合を回避することができる（気象のシミュレーションでは各都市で平均を 0，つまりバイアス成分を取り除き，変動成分とバイアス成分とをわけて学習させている．）．また，他の回避方法としてモジュールの表現能力を限定することも考えられる（周期振動波形，3D 物体のマップでは中間層ユニットを減らすことで，モジュールが複雑な非線形空間を実現しないように制限をかけてある．）．

8.5 ローカルミニマムの問題

MLP-mnSOM は，モジュールを勾配法（つまり BP 法）で学習させるために，ローカルミニマムに陥るモジュールが存在する可能性もある．このローカルミニマムの問題は，マップの連続性を失ったりトポロジー性を破壊するなど重大な問題である．そこで，ローカルミニマムに陥るモジュールをなくすためのアルゴリズムが必要となる．

式 3.9 に示すように，k-th モジュールのタスクはエネルギー関数 $E^k = \sum_i \psi_i^k E_i^k$ を最小化することである．今，モジュール M^k が他のモジュール $M^{k'}$ に一時的に置き換わったとする．もしもエネルギー関数 E^k が置き換えられたことによって改善されるならば，M^k はローカルミニマムに陥っていたと推定できる．その場合，$\boldsymbol{\theta}'$ を $\boldsymbol{\theta}$ にコピーするこ

とでローカルミニマムに陥っている M^k を助け出すことができる．このアルゴリズムを定式化すると次式のようになる．

$$\boldsymbol{\theta}^k(t+1) = \boldsymbol{\theta}^{k'}(t) \quad if \quad \tilde{E}^{kk'} < \beta E^k \tag{8.4}$$

$$\tilde{E}^{kk'} \triangleq \sum_{i=1}^{I} \psi_i^k E_i^{k'} \tag{8.5}$$

ここで，$\tilde{E}^{kk'}$ は一時的に M^k を $M^{k'}$ に置き換えた時のエネルギー関数であり，β は安全項（通常 $0.8-0.9$ 周辺の値をとる）である．*Adaptive process* の前に近傍モジュール間でそのような比較を行うことによって，MLP モジュールはローカルミニマムを回避できると期待できる．そしてこのようなアルゴリズムを採用することによって，モジュール数の増大に従ってローカルミニマムから救い出す機会も増えるため，ローカルミニマムに陥ることが減少するだろう．すなわち，このアルゴリズムは多数の MLP を使うという不利な点を利点に変えるものである．このアルゴリズムは，ローカルミニマム問題を抱える全てのモジュールアーキテクチャに用いることができる．

8.6 一般化されたmnSOMについて

本論文の3章で説明したアーキテクチャは MLP をモジュールとする mnSOM の場合について説明を行った．もちろん，MLP-mnSOM のアーキテクチャは理論的にも明快であり，理解しやすいためだ．しかし，mnSOM のモジュールは MLP だけでなく，他のニューラルネットワークや機能を持たせることができると期待される．そのため本節では，より一般化された mnSOM のアーキテクチャを示す．

今，mnSOM を使うユーザーが I 個のオブジェクト $\mathcal{O} = \{O_1, ..., O_I\}$ に対してマップを作ろうとしているとする．このとき従来の SOM では，各データベクトルがマッピングオブジェクトであり，MLP-mnSOM の場合はそれぞれの関数が1つのオブジェクトに対応する．またオブジェクトの実体が前もって既知である必要ない，すなわち一般的にオブジェクトの実体は未知であると仮定する．その代わり，各オブジェクトから観測されているサンプルデータセットは利用できる．つまり，i-th オブジェクト O_i からデータセット $D_i = \{\boldsymbol{r}_{i1}, ..., \boldsymbol{r}_{iJ}\}$ が観測されているとする．本論文では，$\{\boldsymbol{r}_{ij}\}$ はベクトルデータとする．

mnSOM は K 個の機能モジュール $\{M^1, ..., M^K\}$ から成り，各モジュールは対象となるオブジェクトを再現あるいは模倣する能力を持つようにデザインされたものとする．つまり，モジュールは D_i で学習した後，オブジェクト O_i を近似する能力を持つ．また各機能モジュール M^k の特性はパラメータセット $\boldsymbol{\theta}^k$ で記述されるとする．つまり，MLP-mnSOM の場合，$\boldsymbol{\theta}^k$ は k-th モジュールの結合荷重に相当する．さらに，各機能モ

ジュール M^k はマップ空間で固定された位置 $\boldsymbol{\xi}^k$ を与えられているとする．すなわち $\boldsymbol{\xi}^k$ はマップ空間における M^k の座標であり，$\boldsymbol{\theta}^k$ はデータ空間上における M^k の位置を決定する．

このような状況で，mnSOM は以下のタスクを実施する．

(i) 機能モジュール M^k を適応させることによって観測されたデータセット $\{D_i\}$ からオブジェクト O_i の実体を同定する．

(ii) (i) と同時に，オブジェクト間の類似，相違度を表したマップを生成する．

ここで mnSOM によって生成されたマップはオブジェクトの実体間の関係を表現するものであって，データセット間を直接比較するようなマップではないことに注意して欲しい．

mnSOM を扱うユーザーが気にしなければならないことは，2 つのオブジェクト間あるいはオブジェクトとモジュール間の違いを表すことができる距離測度 $L(O_i, M^k)$ を定義することだけである．そしてこの距離測度を用いることによって 2 つの重要な導関数の定義も決められる．1 つはデータベクトルとモジュール間の誤差の定義である．e_{ij}^k はデータベクトル $\boldsymbol{r}_{ij}$ とモジュール M^k との誤差を表しているとする．このとき，D_i と M^k との平均誤差 E_i^k は O_i と M^k との 2 乗距離の近似になると期待できる．従って，平均誤差 E_i^k は次式で示すように O_i と M^k の推定された距離 $L^2(O_i, M^k)$ として見なせる．

$$L^2(O_i, M^k) \simeq E_i^k \triangleq \frac{1}{J}\sum_{j=1}^{J} e_{ij}^k. \tag{8.6}$$

この誤差の定義は，O_i を実現するための能力を持ったモジュールを採用しているため，通常扱われるモジュールと関連している．つまり，ユーザーの選択したモジュールは D_i を用いて学習することによって E_i^k を最小化すると考えられる．

もう 1 つの導関数は，次式で示す “オブジェクトの内分点” の定義である．

$$\bar{O}(\boldsymbol{m}, \mathcal{O}) \triangleq \arg\min_{O} \sum_{i=1}^{I} m_i L^2(O_i, O). \tag{8.7}$$

ここで $\bar{O}$ は重み $\boldsymbol{m} = \{m_1, ..., m_I\}$ による $\mathcal{O} = \{O_i\}$ の内分点を表す．

mnSOM の一般化されたアルゴリズムも 3 章で説明した MLP-mnSOM の場合と同じように，4 つのプロセスから成り立つ．*Evaluative process* では，O_i と M^k のそれぞれの組み合わせにおける距離を平均誤差 E_i^k の評価で推定する．次の *Competitive,Cooperative process* は MLP-mnSOM の場合と同じである．つまり，*Competitive process* では，式 3.5 によって BMM が決定され，そして *Cooperative process* で内分点の重み ψ_i^k が式 3.6, 3.7 で計算される．最後に，*Adaptive process* で，全てのモジュールは重み $\boldsymbol{\psi}^k(t) =$

$(\psi_1^k(t), ..., \psi_I^k(t))$ によってオブジェクトの内分点となるように更新が行われる．理論的には，M^k は

$$M^k(t+1) = \bar{O}(\boldsymbol{\psi}^k, \mathcal{O}) = \arg\min_M \sum_{i=1}^{I} \psi_i^k(t) L^2(O_i, M). \tag{8.8}$$

のように導入されるだろう．しかし式8.8は $\{O_i\}$ の実体に関する情報が欠如しているために直接評価することはできない．その代わりに，式8.8は式8.6に示すように2乗距離 $L^2(O_i, M^k)$ に平均誤差 E_i^k を代わりとして用いることによって推定することができる．その結果として，更新アルゴリズムは，

$$\boldsymbol{\theta}^k(t+1) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{I} \psi_i^k(t) E_i^k(\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} E^k(\boldsymbol{\theta}) \tag{8.9}$$

として行われる．式8.9における修正はモジュールの学習アルゴリズムに依存するが，モジュールのアルゴリズムがモジュールに D_i が与えられた時に E_i^k を最小化すると仮定しているため，多くの場合，モジュールのタイプに由来するアルゴリズムそのものの修正であろう．最後に，これら4つのプロセスはネットワークが定常状態になるまで繰り返される．

この一般化されたmnSOMのアルゴリズムにおいて，ユーザーはモジュールのアーキテクチャに伴うふさわしい距離測度を選ぶ必要があると先に述べた．これはSOMの骨となるアルゴリズムには触れずに，距離測度によって定義されたオブジェクト空間におけるSOMとして一般化されたmnSOMを見なすためである．

第９章

まとめ

本論文ではモジュラーネットワークと SOM のアーキテクチャを融合させた新しい自己組織化マップとして mnSOM を提案した．そして，mnSOM が工学的に汎用性のあるアーキテクチャ，アルゴリズムとなるように，理論と実践の面から確立を目指した．その結果，さまざまな課題に汎用的に扱える mnSOM のアーキテクチャ，アルゴリズムを理論的に確立できた．さらにシミュレーションによって mnSOM が工学的に有効であることも示した．

我々の mnSOM は Kohonen の SOM を自然な形で拡張したものであり，SOM をより一般的な枠組みで記述したものである．SOM は工学的にさまざまに利用されているため，一般化した SOM である mnSOM はさらに多くの工学的応用が期待できる．

mnSOM の利点の 1 つは，mnSOM の全モジュールが情報処理の能力を持っていることである．このように，mnSOM は機能ごとに規則的に配置された機能モジュールの集合体であると考えられる．このことから，脳の機能地図と mnSOM は対応していると見ることもできる．

以上のことから，今後，私は mnSOM が脳型情報処理モデルとして活躍するよう研究を進める．

研究業績リスト

論文

1 徳永憲洋, 肝付謙二, 古川徹生, 安井湘三, "関数空間型 SOM", 日本神経回路学会誌, vol.12, No. 1, pp.39–51, 2005.

2 K. Tokunaga, T. Furukawa, S. Yasui, "Modular network SOM : Self-organizing maps in function space," *Neural Information Processing - Letter and Reviews*, Vol.8, No.4, pp.15–22, 2005.

国際学会

3 K. Tokunaga, T. Furukawa and S. Yasui, "Modular network SOM: Extension of SOM to the realm of function space," *In Proc. of Workshop on Self-Organizing Maps*, pp.173–178, 2003.

4 K. Tokunaga, T. Furukawa, "Nonlinear ASSOM constituted of autoassociative neural modules," *In Proc. of Workshop on Self-Organizing Masp*, pp.637-644, 2005.

5 T. Furukawa, K. Tokunaga, K. Morishita and S. Yasui, "Modular network SOM(mnSOM): From vector space to function space," *In Proc. of International Joint Conference on Neural Networks*, pp.1581–1586, 2005.

6 S. Kaneko, K. Tokunaga and T. Furukawa, "Modular Network SOM : The architecture, the algorithm and applications to nonlinear dynamical systems," *In Proc. of Workshop on Self-Organizing Maps*, pp.537–544, 2005.

7 K. Tokunaga, S. Taguchi and T. Furukawa, "Realizing the nonlinear adaptive subspace SOM (NL-ASSOM)," *In Proc. of International Conference on Brain-inspired Information Technology*, pp.76, 2005.

8 T. Furukawa, K. Tokunaga, S. Kaneko, K. Kimotsuki and S. Yasui, "Generalized self-organizing maps(mnSOM) for dealing with dynamical systems," *In Proc. of International Symposium on Nonliner Theory and its Applications*, pp.231–234, 2004.

9 T. Furukawa, K. Tokunaga, S. Kaneko, K. Kimotsuki and S. Yasui, "mnSOM : Extension of self-organizing map for mapping function systems and classes," *In Proc. of Postech-Kyutech Joint Workshop*, 2004.

10 K. Tokunaga, K. Kimotsuki, S. Yasui and T. Furukawa, "Self-organizing map of a dynamical system with mnSOM," *In Proc. of Postech-Kyutech Joint Workshop*, 2004.

参考文献

[1] Abe, T., Kanaya, S., Kinouchi, M., Ichiba, Y., Kozuki, T. and Ikemura, T. (2003), Informatics for unvailing hidden genome signatures. *Genome Research*, Vol. 13, pp. 693–702.

[2] Ahalt, A., Krishnamurthy, A. K., Chen, P. and Melton, D. E. (1990), Competitive learning algorithm for vector quantization. *Neural Networks*, Vol. 3, pp. 277–290.

[3] Bishop, C., Svensén, M. and Williams, C. (1998), GTM: The generative topographic mapping. *Neural Computation*, Vol. 10, pp. 215–234.

[4] Deboech, G. and Kohonen, T. (1998), *Visual Explorations in Finance with Self-Organizing Maps*. Springer Finance, London.

[5] Elman, J. L. (1991), Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, Vol. 7, pp. 195–225.

[6] Fritsch, T. (1994), Cellular mobile communication design using self-organizing feature maps. In Yuhas, B. and N. Ansari, e., editors, *Neural Networks in Telecommunications*, pp. 211–232. Kluwer, Boston, MA.

[7] Furukawa, T. (2005), SOM^2 as "SOM of SOMs". In *Proc. of Workshop on Self-Organizing Maps*, pp. 545–552.

[8] Furukawa, T. (2005), SOM of SOMs: Self-organizing map which maps a group of self-organizing maps. In *Proc. of International Conference on Artificial Neural Networks*, Vol. 1, pp. 391–396.

[9] Furukawa, T., Tokunaga, K., Kaneko, S. and Yasui, S. (2004), Generalized self-organizing maps (mnSOM) for dealing with dynamical systems. In *Proc. of International Symposium on Nonliner Theory and its Applications*, pp. 231–234.

[10] Furukawa, T., Tokunaga, K., Morishita, K. and Yasui, S. (2005), Modular network SOM (mnSOM): From vector space to function space. In *Proc. of International Joint Conference on Neural Networks*, pp. 1581–1586.

[11] 古川徹生 (2002), データのクラス振り分けとクラス別モデルの同時推定法. 日本神経

回路学会誌, Vol. 9, No. 2, pp. 92–102.

[12] Gersho, A. and Gray, R. M. (1992), *Vector quantization and signal compression*. Kluwer, Boston, MA.

[13] Haritopoulos, M., Yin, H. and Allinson, N. M. (2002), Image denoising using self-organizing map-based nonlinear independent component analysis. *Neural Networks*, Vol. 15, pp. 1085–1098.

[14] Ishii, K., Nishida, S. and Ura, T. (2004), A self-organizing map based navigation system for an underwater vehicle. In *Proc. of International Conference on Robotics and Automation*, pp. 4466–4471.

[15] Jabcobs, R., Jordan, M., Nowlan, J. and Hinton, G. (1991), Adaptive mixtures of local experts. *Neural Computation*, Vol. 2, pp. 79–87.

[16] Jin, H., Shum, W.-H., Leung, K.-S. and Wong, M.-L. (2004), Expanding self-organizing map for data visualization and cluster analysis. *Information Sciences*, Vol. 163, pp. 157–173.

[17] Jordan, M. and Jacobs, R. (1994), Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, Vol. 6, pp. 181–214.

[18] Kaneko, S., Tokunaga, K. and Furukawa, T. (2005), Modular network SOM: The architecture, the algorithm and applications to nonlinear dynamical systems. In *Proc. of Workshop on Self-Organizing Maps*, pp. 537–544.

[19] Kangas, J. (1994), *On the analysis of pattern sequences by self-organizing maps*. PhD thesis, Helsinki University of Technology, Espoo, Finland.

[20] Kaski, S., Nikkilä, J., Oja, M., Venna, J., Törönen, P. and Castrén, E. (2003), Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics*, Vol. 4, pp. 48–61.

[21] 川人光男 (2001), 脳の計算理論, pp. 18–23. 産業図書.

[22] Kirkland, W. R. and Taylor, D. P. (1994), Neural network channel equalization. In Yuhas, B. and N. Ansari, e., editors, *Neural Networks in Telecommunications*, pp. 141–171. Kluwer, Boston, MA.

[23] Kohonen, T. (1993), Generalization of the self-organizing map. In *Proc. of International Joint Conference on Neural Networks*, pp. 457–462.

[24] Kohonen, T. (2001), *Self-organizing maps*. Springer-Verlag.

[25] Kohonen, T., Kaski, S. and Lappalainen, H. (1993), Self-organized formation of various invariant-feature filters in the adaptive-subspace SOM. *Neural Computation*, Vol. 9, pp. 1321–1344.

[26] Kramer, M. A. (1991), Nonlinear principal component analysis using autoassociative

neural networks. *AIChE journal*, Vol. 37, No. 2, pp. 233–243.

[27] Kurimo, M. (2002), Thematic indexing of spoken documents by using self-organizing maps. *Speech Commun.*, Vol. 38, No. 1, pp. 29–45.

[28] Linde, Y., Buzo, A. and Gray, R. M. (1980), An algorithm for vector quantizer design. *IEEE Transactions on Communications*, Vol. COM-28, pp. 84–95.

[29] Luttrell, S. P.Derivation of a class of training algorithms. *IEEE Transaction Neural Networks*, Vol. 1, p. 229.

[30] Luttrell, S. P. (1989), Self-organization: A derivation from first principles of a class of learning algorithms. In *Proc. of International Joint Conference on Neural Networks*, Vol. 1, pp. 495–498, DC, Washington.

[31] Ma, F. and Xia, S. (1998), A multiscale approach to automatic medical image segmentation using self-organizing map. *Journal of Computer Science and Technology*, Vol. 13, No. 5, pp. 402–409.

[32] MacQueen, J. (1967), Some methods for classification and analysis of multivariate observations. In *Proc. of Fifth Berkeley Symp. on Math. Stat. and Prob.*, Vol. 1, pp. 281–296.

[33] Markey, M. K., Tourassi, G. D. and Floyd, C. E. (2003), Self-organizing map for cluster analysis of a breast cancer database. *Artificial Intelligence in Medicine*, Vol. 27, pp. 113–127.

[34] Martinetz, T., Ritter, H. and Schulten, K.Three-dimentional neural net for learning visuo-motor coordination of a ronbot arm. *IEEE Trans. on Neural Networks*, Vol. 1, p. 131.

[35] 森下健司, 徳永憲洋, 安井湘三, 古川徹生 (2005), モジュラーネットワーク型 SOM による 3 次元物体集合の自己組織化マップ. 電子情報通信学会技術報告書ニューロコンピューティング, No. 210, pp. 99–104.

[36] Motter, M. A. (2000), Predictive multiple model switching control with the self-organizing map. In *Proc. of International Joint Conference on Neural Networks*, Vol. 4, pp. 4317–4322, Como, Italy.

[37] Oja, E. (1991), Data compression, feature extraction, and autoassociation in feedforward neural networks. In *Proc. of International Conference on Artificial Neural Networks*, pp. 737–745.

[38] Oja, E. (1991), Data compression, feature extraction, and autoassociation in feedforward neural networks. In *Proc. of International Conference on Artificial Neural Networks*, Vol. 1, pp. 737–745.

[39] Penfield, W. and Boldrey, E. (1937), Somatic motor and sensory representation in the

cerebral cortex of man as studied by electrical stimulation. *Brain*, Vol. 37, pp. 389–443.

[40] 齊藤光俊, 古川徹生, 安井湘三 (2003), 競合モジュラーネットワークによる気象データ分類とモデル推定. 電子情報通信学会技術報告ニューロコンピューティング, No. 130, pp. 85–89.

[41] Suzuki, S. and Ando, H. (2000), A modular network scheme for unsupervised 3d object recognition. *Neurocomputing*, Vol. 31, pp. 15–28.

[42] 鈴木敏, 安藤広志 (1996), 2 次元射影像からの 3 次元物体の認識と類別-モジュール構造を用いた教師なし学習モデル-. 電子情報通信学会論文誌, Vol. J79-D-II, No. 7, pp. 1291–1300.

[43] Tokunaga, K. and Furukawa, T. (2005), Nonlinear ASSOM constituted of autoassociative neural modules. In *Proc. of Workshop on Self-Organizing Maps*, pp. 637–644.

[44] Tokunaga, K., Furukawa, T. and Yasui, S. (2003), Modular network SOM: Extension of SOM to the realm of function space. In *Proc. of Workshop on Self-Organizing Maps*, pp. 173–178.

[45] Tokunaga, K., Furukawa, T. and Yasui, S. (2005), Modular network SOM: Self-organizing maps in function space. *Neural Information Processing - Letter and Reviews*, Vol. 8, No. 4, pp. 15–22.

[46] 徳永憲洋, 肝付謙二, 古川徹生, 安井湘三 (2005), 関数空間型 SOM. 日本神経回路学会誌, Vol. 12, No. 1, pp. 39–51.

[47] Vesanto, J. (1999), SOM-based data visualization methods. *Intelligent Data Analysis*, Vol. 3, pp. 111–126.

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

[49] Zeller, M., Sharma, R. and Schulten, K. (1997), Motion planning of a pneumatic robot using a neural network. *Control Systems Magazine*, Vol. 17, No. 3, pp. 89–98.

[50] Zhang, B., Fu, M. and Jabri, M. (1999), Handwritten digit recognition by adaptive-subspace self-organizing map (ASSOM). *IEEE Trans. Neural Networks*, Vol. 10, No. 4, pp. 939–945.

謝辞

本論文の執筆に当たり以下の方々にお世話になりました．この場を借りてお礼申し上げます．

九州工業大学大学院生命体工学研究科脳情報専攻の安井湘三教授には私が学部の頃から，研究だけでなく公私にわたってご指導，ご鞭撻を賜りました．また本論文の主体となる mnSOM について，沢山の助言と指導をいただきました．安井教授のおかげで本論文を完成することができました．安井教授に対しては本当に感謝の念が絶えません．心よりお礼を申し上げると共に，今後自らの未熟さについて一層精進していきたいと思います．

九州工業大学大学院生命体工学研究科脳情報専攻の古川徹生助教授と私は mnSOM が提案された当時から，mnSOM はきっと素晴らしいモデルとなるだろう，と信じて研究を続けきました．そしてついに mnSOM は理論的にも確立され，さらに脳情報処理モデルとしても確立したものとなりました．古川助教授には研究だけでなく，公私にわたって多くのご指導と助言をいただきました．その中で研究に対する考え方，研究をするということの厳しさ，楽しさ，研究とは何かを学ばせていただきました．古川教授には本当に感謝の念が絶えません．心よりお礼を申し上げます．

安井研究室に所属し，mnSOM に関して研究をしてくれた，肝付謙二君，森下健司君，森山健一郎君（３人とも現在 OB）には，ダイナミカルシステム集合，周期振動波形集合そして 3D 物体集合の mnSOM によるマッピングに関してシミュレーションをしていただきました．また，これらシミュレーションの過程で，本研究のクオリティを上げるコメントや考察をしていただきました．おかげで，このような論文を書くことができました．感謝と共にお礼を申し上げます．また古川研究室のメンバーの皆さんには，この場をお借りしてお礼申し上げます．

この研究をする際に，支えてくれた家族，そして友人に感謝いたします．

なお，この学位論文の研究の一部は，21 世紀 COE プログラム「生物とロボットが織りなす脳情報工学の世界」（拠点番号 J19）の推進事業として実施いたしました．関係者各位ならびに関係部署に深く感謝いたします．

さらに本研究の一部は文部科学省科学研究費補助金（No. 17500193）の補助を得て行われたものである．

新しい自己組織化マップの創出 ：
モジュラーネットワーク SOM

著 者: 徳永 憲洋

発 行: 2006 年 3 月

学 位: 博士（工学）