修士学位論文

# クラスタリングを利用した<br>語義判別規則の教師なし学習

平成14年度

茨城大学大学院理工学研究科

システム工学専攻

高橋篤史

# 要旨

本論文では教師なし学習を実現するためにクラスタリングの一種である2つの手法に関して論じる．1つはEMアルゴリズムを用いた手法であり，もう1つはファジィクラスタリングを用いた手法である．実験では多義語の曖昧性解消問題を取り上げ，提案手法の評価を行う．

自然言語処理では個々の問題を分類問題として定式化し，帰納学習の手法を利用して，その問題を解決するというアプローチが大きな成功をおさめている．しかしこのアプローチには帰納学習で必要とされる訓練データを用意しなければならないという大きな問題がある．近年，この問題に対する1つの対処法として少量のラベル付き訓練データから得られる分類規則の精度を，大量のラベルなし訓練データによって高めてゆく教師なし学習が提案されている．

教師なし学習を実現するための1つの方法としてクラスタリングが考えられる．イメージ的にはラベル付きの訓練事例を各クラスターとして捉え，クラスタリングの手法を用いてラベルなしの訓練事例をクラスターに帰属させてゆく．ここではクラスタリングの手法として EM アルゴリズムとファジークラスタリングを用いて教師なし学習を試みる．

EM アルゴリズムは一種のクラスタリング手法ではあるが，通常は，部分的に欠損値のある不完全な観測データ $x_1, x_2, \cdots, x_N$ から，そのデータを発生する確率モデル $P_\theta(x)$ を推定する手法として扱われている．$P_\theta(x)$ は未知パラメータ $\theta$ を含み，$P_\theta(x)$ の推定とは，$\theta$ の推定に帰着される．分類問題の教師なし学習では，ラベル付き訓練データが完全な観測データ，ラベルなし訓練データがラベルを欠損値とした不完全な観測データとなる．EM アルゴリズムは，現時点での $\theta$ を使って，モデル $P_\theta(c|x_i)$ のもとでの $logP_{\hat{\theta}}(x_i, c)$ の期待値を取る（E-step）．次に，この期待値を最大にするような $\hat{\theta}$ を求める（M-Step）．$\hat{\theta}$ を新たな $\theta$ として先の E-step と M-step を繰り返す．ここで $c$ は欠損値となるラベルである．$P_\theta(x)$ を Naive Bayes のモデル，$\theta$ をラベル $c_i$ のもとで素性 $f_k$ が起る条件付き確率 $p(f_k|c_i)$ に設定することで，$P_\theta(c|x_i)$ と $logP_{\hat{\theta}}(x_i, c)$ が具体的な形で与えられ，計算が可能となる．

ファジィクラスタリングは，個体がクラスに帰属する度合いに曖昧さを認めるという考えに基づいている．曖昧さの表現はファジィ理論による要素への帰属度によって表される．帰属度を求めるために2次の乗パラメータを持つ目的関数とc-平均法を用いる．ファジィクラスタリングを教師なし学習として利用するためには，まずラベル付きの各訓練事例をクラスに設定し，次にファジィクラスタリングを用いて各ラベルなしの事例の各クラス（ラベル付きの訓練事例）への帰属度を求める．この処理により各クラスのプロトタイプの位置がより適切な位置へ移動する．実際の識別は，移動された後のクラスのプロトタイプの集合を訓練事例として，k-最近傍法により行うことができる．

実験ではSENSEVAL2の日本語語義タスクを題材にした．日本語辞書タスクは単純な多義語の曖昧性解消問題であり，分類問題として扱える．実験の結果，EMアルゴリズムを用いた教師なし学習とファジィクラスタリングを行った教師なし学習のどちらの手法においてもラベル付きの訓練データのみからの学習手法の精度を上回ることは出来なかった．ただし個別の単語でみると大きく正解率が改善される単語も多かった．

ラベルなしデータを用いると精度が悪くなる問題を予め検知し，精度向上が行える問題にだけ教師なし学習を適用するのが，教師なし学習手法の現実的な利用方法だと思われる．そのような検知方法が今後の課題である．

# Abstract

In this master's thesis, I describe two clustering methods to implement unsupervised learning. One uses the EM algorithm, and another uses the fuzzy clustering. In experiments, I apply these method to word sense disambiguation problems to evaluate them.

Many problems in natural language processing can be converted into classification problems, and be solved by an inductive learning method. This strategy has been very successful, but it has a serious problem in that an inductive learning method requires labeled data. To overcome this problem, unsupervised learning methods using huge unlabeled data to boost the performance of rules learned by small labeled data have been proposed recently.

Clustering is a way to implement unsupervised learning. Intuitively speeking, this method regards each labeled data as a cluster, and attaches unlabeled data to clusters by using a clustering method. In this thesis, I use the EM algorithm and the fuzzy clustering as the clustering method.

The EM algorithm is a kind of clustering methods, but is generally regarded as the method to estimate the probability model $P_\theta(x)$ from observed data $x_1, x_2, \cdots, x_N$ with missing values. Since $P_\theta(x)$ includes unknown parameter $\theta$, the estimation of $P_\theta(x)$ means the estimation of $\theta$. In classification problems, labeled data is correspond to complete data, and unlabeled data is correspond to uncompleted data with unknown labels as missing values. First, EM algorithm computes the mean of $logP_{\hat{\theta}}(x_i, c)$ under the model $P_\theta(c|x_i)$ by using current $\theta$, that is E-step. Next, it computes $\hat{\theta}$, which gives the max of the mean of $logP_{\hat{\theta}}(x_i, c)$, that is M-step. Last, the $\theta$ is updated to this $\hat{\theta}$, and then E-step and M-step are iterated. In my method, $c$ is a label, and handled as a missing value. Furthermore, Naive Bayes model is set as $P_\theta(x)$, and $p(f_k|c_i)$, which means generating probability of the feature $f_k$ under the label $c_i$ is set as $\theta$. By this setting, $P_\theta(c|x_i)$ and $logP_{\hat{\theta}}(x_i, c)$ come to be computable.

The Fuzzy clustering allows an object to belong plural classes with the degree to belong the class. In the Fuzzy clustering, this degree is defined as the membership degree. In this thesis, the membership degree is computed by using the general target function and the fuzzy c-means algorithm. In order to implement unsupervised learning by the Fuzzy clustering, each labeled training data is regared as each class, and then the membership degree of each unlabeled data to classes is computed by the Fuzzy clustering. The prototypes of classes are moved by this procedure. In real classification, we use k-nearest neighbor method by regarding the moved prototypes as labeled training data.

In experiment, I used word sense disambiguation problems taken in the Japanese Dictionary Task in SENSEVAL2. This task can be regared as the classification problem because of this is simple word sense disambiguation problems. In experiments, both of the EM algorithm and the Fuzzy clustering could not outperform the general learning method using only labeled training data. However, these methods improved the precisions of some words dramatically.

I believe that unsupervised method should be used for such problems that unsupervised method is useful for. The future work is to study how to detect the usefulness of unsupervised method for the target problem in advance.

# 目次

# 表目次

# 図目次

# 第1章　序論

## 1.1　はじめに

自然言語処理では個々の問題を分類問題として定式化し，帰納学習の手法を利用して，その問題を解決するというアプローチが大きな成功をおさめている．しかしこのアプローチには帰納学習で必要とされる訓練データを用意しなければならないという大きな問題がある．

近年，この問題に対する1つの対処法として少量のラベル付き訓練データから得られる分類規則の精度を，大量のラベルなし訓練データによって高めてゆく教師なし学習が提案されている．

ここではクラスター分析の手法であるEMアルゴリズム [3] とファジィクラスタリング [9] [11] の2つの手法を利用した教師なし学習を試みる．

EMアルゴリズムを用いた手法は，Naive BayesとEMアルゴリズムを組み合わせた手法である．本質的には，あるクラス $c$ のもとである素性 $f$ が発生する確率 $P(f|c)$ を求める．Naive Bayesのモデルが使えれば，この確率から分類器を作成できる．ラベル付きの訓練事例から $P(f|c)$ は簡単に計算できるが，ラベル付き訓練事例が少量の場合，$P(f|c)$ の信頼性は低い．そこでラベルなし訓練事例を用いて $P(f|c)$ の信頼性を高める．これは全体のデータの発生確率が最大になるように $P(f|c)$ を設定すればよい．この計算にEMアルゴリズムが利用される．

ファジィクラスタリングは，個体がクラスに帰属する度合いに曖昧さを認めるという考えに基づいている．曖昧さの表現はファジィ理論による要素への帰属度によって表される．ここでは，まずラベル付きの各訓練事例をクラスに設定し，次にファジィクラスタリングを用いて各ラベルなしの事例の各クラス（ラベル付きの訓練事例）への帰属度を求める．この処理により各クラスの代表点の移動が起る．実際の識別は，移動された後のクラスの代表点の集合を訓練事例として，k-最近傍法により行う．

実験ではSENSEVAL2の日本語語義タスク [5] を題材にした．名詞50単語を用いた実験でのNaive Bayesによる結果は76.78%であり，EMアルゴリズムを用いた学習では

72.82%であった [7]. また，通常の k-最近傍法 ($k = 5$) による結果は，76.83%であり，ファジィクラスタリングを用いた教師なし学習では，76.07%であった [8]. 両手法とも全体的には精度の向上はならなかったが，個別に見ると精度が向上している単語も多かった．有効な利用方法などを考察する．

## 1.2　本論文の構成

まず，第2章でEMアルゴリズムを用いた教師なし学習について述べる．第3章で，ファジィクラスタリングを用いた教師なし学習について述べる．続いて第4章で両手法を多語義の語義曖昧性解消問題に適用した実験およびその結果について説明し，第5章で本研究の考察を述べる．

また巻末には付録として，各単語の正解率のグラフと両手法のプログラムのソースリストを添付した．

# 第2章　EMアルゴリズム

EMアルゴリズムを利用した手法は，Naive Bayes と EMアルゴリズムを組み合わせた手法である．本質的には，あるクラス $c$ のもとである素性 $f$ が発生する確率 $P(f|c)$ を求める．Naive Bayes のモデルが使えれば，この確率から分類器を作成できる．

ラベル付きデータから $P(f|c)$ は簡単に計算できるが，ラベル付きデータが少量の場合，$P(f|c)$ の信頼性は低い．そこでラベルなしデータを用いてより $P(f|c)$ の信頼性を高める．これは全体のデータの発生確率が最大になるように $P(f|c)$ を設定すればよい．この計算に EM アルゴリズムが利用される．この手法も，本質的に，複数観点を利用している．事例は素性のベクトル $(f_1, f_2, \cdots, f_n)$ として表されるので，ある素性 $f_i$ から，その事例のクラス $c$ が判別できたとき，$P(f_j|c)$ の精度を高められるからである．この手法は Naive Bayes のモデルが有効な，文書分類に応用されている．

## 2.1　基本的考え方

2つの言語モデル $P_A(x)$，$P_B(x)$ から，$P_A(x)$ あるいは $P_B(x)$ を確率的に選択するような言語モデル $P(x)$ を構成する問題を考える．ここで，$P_A(x)$ が選ばれる確率を $\lambda$ とすると，言語モデル $P(x)$ は次のように表すことができる．

$$P(x) = \lambda P_A(x) + (1-\lambda)P_B(x) \tag{2.1}$$

なお，このようなモデルを混合モデルと呼ぶ．

ここで問題となるのは，実際に観測されたデータ $x_1, \ldots, x_N$ から，未知パラメータ $\lambda$ の値を求めることである．もし，各 $x_i$ について，$P_A$ あるいは $P_B$ のどちらのモデルが選ばれるかを知ることができれば問題は非常に簡単である．$x_1, \ldots, x_N$ の生成において，$P_A$ が選ばれた回数を $N_A$ とすれば，$\lambda = N_A/N$ と求めることができる．しかし，どちらのモデルが選ばれたかは，一般に知ることができない．すなわち，$P_A$ あるいは $P_B$ のどちらが選ばれたかというモデルの内部状態はモデルの内部に隠れており，観測データ $x_1, \ldots, x_N$ だけからは決定することができない．このようなデータ $x_1, \ldots, x_N$ を不完全

データ（incomplete data）と呼ぶ．これに対し，内部状態が既知であるデータを完全データ（complete data）と呼ぶ．いまの場合，$x_i$ の生成において，どちらのモデルが使われたかを $q_i$ により示すことにすると，完全データは $(x_1, q_1), \ldots, (x_N, q_N)$ ということになる．

EMアルゴリズムは，以下のようにして未知パラメータの値を推定する．まず，パラメータ $\lambda$ に適当な値を仮定する．次に，この仮定した値のもとで，各観測データ $x_i$ の生成にモデル $P_A$ が使われた回数の期待値 $E_\lambda[A|x_i]$ を（最尤推定により）求める．

$$\begin{aligned} E_\lambda[A|x_i] &= \frac{\text{モデル } P_A \text{により } x_i \text{が生成された確率}}{\text{いずれかのモデルにより } x_i \text{が生成された確率}} \\ &= \frac{\lambda P_A(x_i)}{\lambda P_A(x_i) + (1-\lambda) P_B(x_i)} \end{aligned} \tag{2.2}$$

観測データ全体の生成にモデル $P_A$ が使われた回数の期待値は $\sum_i E_\lambda[A|x_i]$ として求めることができる．これを観測データの個数で平均化することにより，パラメータ $\lambda$ の値を更新する．更新された $\lambda$ の値を $\bar{\lambda}$ とすると，$\bar{\lambda}$ は次のようになる．

$$\bar{\lambda} = \frac{1}{N} \sum_{i=1}^{N} \frac{\lambda P_A(x_i)}{\lambda P_A(x_i) + (1-\lambda) P_B(x_i)} \tag{2.3}$$

以上の手続きを繰り返すことにより，パラメータ $\lambda$ は最適な値に近づいていくと期待できる．

## 2.2　EMアルゴリズム

最初に，EMアルゴリズムが用いられる一般的な状況について述べる．$x_1, \ldots, x_N$ をモデル $P_\theta(x)$ から得られた観測データとする．問題は，観測データ $x_1, \ldots, x_N$ から，未知パラメータ $\theta$の値を推定することである．ただし，$x_1, \ldots, x_N$ は不完全データであり，各 $x_i$ を生成した内部状態を一意的に決定することはできない．

EMアルゴリズムは，上のような問題に対して，繰り返しにより，逐次，モデルの対数尤度を増加させるようなパラメータを推定する方法である．すなわち，すでに求められたパラメータ $\theta$ から，モデルの対数尤度を増加させるような新しいパラメータ値 $\bar{\theta}$ を推定するという操作を繰り返すことにより，モデルの対数尤度を最大化するパラメータを求める．

ここで，観測データ $x_i$ について，パラメータ $\theta$ を $\bar{\theta}$に更新したときの対数尤度の差を求める．なお，以下では，モデルの内部状態を表す変数を $y$ とする．

$$\begin{aligned}
\log P_{\bar{\theta}}(x_i) - \log P_\theta(x_i) &= \log \frac{P_{\bar{\theta}}(x_i)}{P_\theta(x_i)} \\
&= \sum_y P_\theta(y|x_i) \log \frac{P_{\bar{\theta}}(x_i)}{P_\theta(x_i)} \\
&= \sum_y P_\theta(y|x_i) \log \left[ \frac{P_{\bar{\theta}}(x_i, y)}{P_\theta(x_i, y)} \frac{P_\theta(y|x_i)}{P_{\bar{\theta}}(y|x_i)} \right] \\
&= \sum_y P_\theta(y|x_i) \log \frac{P_{\bar{\theta}}(x_i, y)}{P_\theta(x_i, y)} \\
&\quad + \sum_y P_\theta(y|x_i) \log \frac{P_\theta(y|x_i)}{P_{\bar{\theta}}(y|x_i)} \qquad (2.4)
\end{aligned}$$

ここで，

$$Q(\theta, \bar{\theta}) = \sum_y P_\theta(y|x_i) \log P_{\bar{\theta}}(x_i, y) \qquad (2.5)$$

とおく．また，ジェンセンの不等式より，次が成り立つ．

$$\sum_y P_\theta(y|x_i) \log \frac{P_\theta(y|x_i)}{P_{\bar{\theta}}(y|x_i)} \geq 0 \qquad (2.6)$$

したがって，

$$\log P_{\bar{\theta}}(x_i) - \log P_\theta(x_i) \geq Q(\theta, \bar{\theta}) - Q(\theta, \theta) \qquad (2.7)$$

式（2.7）より，$Q(\theta, \bar{\theta}) > Q(\theta, \theta)$ となる $\bar{\theta}$ を見つけることにより，データ $x_i$ に対する対数尤度を増加させることができる．また，対数尤度を最も増加させるためには，$Q(\theta, \bar{\theta})$ を最大にするような $\bar{\theta}$ を求めればよい．以上をまとめると，EMアルゴリズムによるパラメータ推定アルゴリズムは次のようになる．

〈EMアルゴリズム〉

1. $\theta$ に適当な初期値を与える．

2. $\theta$ が収束するまで，次のEステップとMステップを交互に繰り返す．
   Eステップ：$Q(\theta, \bar{\theta})$ を計算する．
   Mステップ：$\theta = arg \max_{\bar{\theta}} Q(\theta, \bar{\theta})$ により $\theta$ を更新する．

EMアルゴリズムの意味するところは，モデル $P_\theta(y|x_i)$ のもとでの $\log P_{\bar{\theta}}(x_i, y)$ の期待値をとり，この期待値を $\bar{\theta}$ の関数とみなして最大化することにあり，モデルの対数尤度を単調に増加させることができるということである．なお，EMアルゴリズムは一般に局所的最大値に収束するにすぎない．どのような値に収束するかは初期値の与え方に大きく依存するので，複数の初期値で試すなどの工夫が必要である．

また，EMアルゴリズムを複数の観測データ $x_1, \ldots, x_N$ に適用するためには，$Q$ 関数を次のように定義すればよい．

$$Q(\theta, \bar{\theta}) = \sum_{i=1}^{N} \sum_{y} P_\theta(y|x_i) \log P_{\bar{\theta}}(x_i, y) \tag{2.8}$$

**例．混合モデルの推定**

$P_1(\cdot), \ldots, P_M(\cdot)$ から得られる混合モデル

$$P(x) = \sum_{j=1}^{M} \lambda_j P_j(x) \quad (\text{ただし}, \sum_{j=1}^{M} = 1)$$

に対して，観測データ $x_1, \ldots, x_N$ から $\lambda_j$ の値を推定する問題を考える．$P_\lambda(x_i, j) = \lambda_j P_j(x_i)$ にEMアルゴリズムを適用すると $Q$ 関数は次のように定義される．

$$Q(\lambda, \bar{\lambda}) = \sum_{i=1}^{N} \sum_{j=1}^{M} P_\lambda(j|x_i) \log P_{\bar{\lambda}}(x_i, j)$$

ここで，ラグランジュの未定係数法を用いて，$Q$ 関数を最大化するような $\bar{\lambda}_j$ を求める．制約条件 $\sum_j \bar{\lambda}_j = 1$ に対するラグランジュ乗数を $\gamma$ とすると，ラグランジュ関数は次のようになる．

$$\mathcal{L}(\bar{\lambda}, \gamma) = Q(\lambda, \bar{\lambda}) - \gamma \left[ \sum_{j=1}^{M} \bar{\lambda}_j - 1 \right]$$

変数 $\bar{\lambda}_j$ で偏微分すると，

$$\frac{\partial \mathcal{L}}{\partial \bar{\lambda}_j} = \sum_{i=1}^{N} P_\lambda(j|x_i) \frac{1}{\bar{\lambda}_j} - \gamma$$

を得る．上式を0とおいて，$\bar{\lambda}_j$ について解くと，

$$\bar{\lambda}_j = \frac{1}{\gamma}\sum_{i=1}^{N} P_\lambda(j|x_i)$$

なお，$\sum_j \bar{\lambda}_j = 1$ より $\gamma = N$ となることが導かれる．また，$P_\lambda(j|x_i)$ は以下のように計算することができる．

$$\begin{aligned} P_\lambda(j|x_i) &= \frac{P_\lambda(x_i, j)}{P_\lambda(x_i)} \\ &= \frac{\lambda_j P_j(x_i)}{\sum_{j=1}^{M} \lambda_j P_j(x_i)} \end{aligned}$$

したがって，

$$\bar{\lambda}_j = \frac{1}{N}\sum_{i=1}^{N} \frac{\lambda_j P_j(x_i)}{\sum_{j=1}^{M} \lambda_j P_j(x_i)}$$

## 2.3　Naive Bayes による語義判別

ある事例 $x$ が素性のベクトルとして，以下のように表現されたとする．

$$x = (f_1, f_2, \cdots, f_n)$$

$x$ の分類先のクラスの集合を $C = \{c_1, c_2, \cdots, c_m\}$ と置く．問題は $P(c|x)$ の分布を推定することである．実際に，$x$ のクラス $c_x$ は以下の式で求まる．

$$c_x = arg\max_{c \in C} P(c|x)$$

ベイズの定理を用いると，

$$P(c|x) = \frac{P(c)P(x|c)}{P(x)}$$

なので，結局，以下が成立する．

$$c_x = arg\max_{c \in C} P(c)P(x|c)$$

ここで，$P(c)$ は比較的簡単に推定できる．問題は，$P(x|c)$ の推定だが，これは現実的には難しい．Naive Bayes のモデルは，この推定に以下の仮定を導入する．

$$P(x|c) = \prod_{i=1}^{n} P(f_i|c) \tag{2.9}$$

$P(f_i|c)$の推定は比較的容易であるために，結果として$P(x|c)$が推定できる．Naive Bayesを使った分類がうまくゆくかどうかは，式2.9の仮定をできるだけ満たすような素性を選択することである．文書分類であれば，各素性を各単語の生起に設定することで，Naive Bayesが有効であることが知られている．

語義判別問題でも式2.9の仮定をできるだけ満たすような素性を選択すればNaive Bayesが利用できる．

本論文では (e1) 直前の単語，(e2) 直後の単語，(e3) 前方の内容語（2つまで），(e4) 後方の内容語（2つまで），の4つの素性を設定した．例えば，「声」の語義は『意見』という語義と『喉から発声される音』という語義がある．また，「日本国民の声を集めました」という文は以下のように形態素解析される．各行が分割された単語であり，第1列が表記，第2列が原型，第3列が品詞を表す．

| | | |
|---|---|---|
| 日本 | 日本 | 名詞-固有名詞-地域-国 |
| 国民 | 国民 | 名詞-一般 |
| の | の | 助詞-連体化 |
| 声 | 声 | 名詞-一般 |
| を | を | 助詞-格助詞-一般 |
| 集め | 集める | 動詞-自立 |
| まし | ます | 助動詞 |
| た | た | 助動詞 |

この結果から以下の4つの素性ができる．

e1=の，e2=を，
e3 ={日本，国民}，e4={ 集める }

e3とe4の素性は集合になるが，学習の際に以下のように分割する．

e3=日本，　e3=国民，　e4=集める

以上のように設定した素性集合がNaive Bayesが仮定する式2.9をどの程度満たすかはわからない．ただ，名詞の場合，その単語の右文脈と左文脈はほぼ独立と考えて良い点と，2単語列を素性として含めない，などを考慮して設定した．

## 2.4　EMアルゴリズムによる教師なし学習

分類問題の解決にNaive Bayesが使えれば，Nigamらが提案した教師なし学習が利用できる．そこではEMアルゴリズムを用いることで，ラベルなしデータを用いて，ラベ

ル付きデータから学習された分類器の精度を向上させる．なお，ここでのラベル付きデータは上で述べた完全データであり，ラベルなしデータは不完全データに該当する．

以下にポイントとなる式とアルゴリズムを示す [3].

基本となるのは，あるクラス $c_j$ のもとで，素性 $f_i$ が発生する確率 $P(f_i|c_j)$ を求めることである．これは以下の式で求まる．この式は頻度 0 の部分を考慮したスムージングを行っている．

$$P(f_i|c_j) = \frac{1 + \sum_{k=1}^{|D|} N(f_i, d_k) P(c_j|d_k)}{|F| + \sum_{m=1}^{|F|} \sum_{k=1}^{|D|} N(f_m, d_k) P(c_j|d_k)} \tag{2.10}$$

式 2.10 の $D$ はラベル付けされたデータとラベル付けされていないデータを合わせた訓練データ全体を示す．$D$ の各要素を $d_k$ で表す．$F$ は素性全体の集合である．$F$ の各要素を $f_m$ で表す．また，$N(f_i, d_k)$ は，訓練事例 $d_k$ に含まれる素性 $f_i$ の個数を表す．ここでの設定では，$N(f_i, d_k)$ は 0 か 1 の値であり，ほとんどの場合 0 である．$P(c_j|d_k)$ は訓練データがクラス $c_j$ を持つ確率である．ラベル付けされたデータに対しては，0 か 1 の値をとる．ラベル付けされていないデータに対して，最初は 0 であるが，EM アルゴリズムの繰り返しによって，徐々に適切な値に更新されて行く．

式 2.10 を利用して，以下の分類器が作成できる．

$$P(c_j|d_i) = \frac{P(c_j) \prod_{f_n \in K_{d_i}} P(f_n|c_j)}{\sum_{r=1}^{|C|} P(c_r) \prod_{f_n \in K_{d_i}} P(f_n|c_r)} \tag{2.11}$$

ここで，$C$ はクラスの集合である．$K_{d_i}$ は訓練事例 $d_i$ に含まれる素性の集合を示す．$P(c_j)$ はクラス $c_j$ の発生確率であり，以下の式で計算できる．

$$P(c_j) = \frac{1 + \sum_{k=1}^{|D|} P(c_j|d_k)}{|C| + |D|}$$

EM アルゴリズムは式 2.10 を利用して，$P(f_i|c_j)$ を求め (E-step)，次のこの値を利用して式 2.11 から分類器を作成し，ラベル付けされていない事例 $d_i$ に対して，$P(c_j|d_i)$ を求める (M-step)．この E-step と M-step を交互に繰り返して，$P(f_i|c_j)$ と $P(c_j|d_i)$ を収束するまで更新してゆく．最終的には式 2.11 から分類が行える．

# 第3章　ファジィクラスタリング

## 3.1　基本的事項

ここではクラスタリングの対象となる個体を記号 $x_1, x_2, \ldots, x_n$ で表すことにする．また，個体の集まりを記号 $X$ で表す．このとき，$X$ が要素 $x_1, x_2, \ldots, x_n$ で構成されていることを，$X = \{x_1, x_2, \ldots, x_n\}$ と表すことができる．個体の数は一般に $n$ 個と仮定する．個体のことを対象（subject）と呼ぶこともある．また，個体を $x, y$ のように添字を付けずに表すことがある．さらに，集合 $X$ の要素であることを明示するために $x, y \in X$ と書くこともある．

さて，個体 $x_k (k = 1, \ldots, n)$ は単なる記号の場合もあるが，個体に関するデータと同一視されることが多い．たとえばA君の身長，体重，胸囲がそれぞれ 167.0，55.0，85.0 であるとし，A君を含む個体の集合をクラスタリングするとすれば，個体であるA君をベクトル $A = (167.0, 55.0, 85.0)$ と同一視するのが便利である．一般に，$x_k$ に関して $p$ 個の実変数が存在し，それらのデータが与えられている場合が最も頻繁に考察されている．このとき，$x_k^1, x_k^2, \ldots, x_k^p$ が個体 $x_k$ に関するデータであるとし，個体を $p$ 次元実ベクトル $x_k = (x_k^1, x_k^2, \ldots, x_k^p)^T$ と同一視する．$T$ は転置を示す記号であり，したがって $x_k$ は列ベクトルである．$p$ 次元ベクトル空間を $R^p$ と書き，個体をこの空間の点

$$x_k = (x_k^1, x_k^2, \ldots, x_k^p)^T \in R_p, k = 1, \ldots, n$$

とみなす．このとき $X$ は有限個の点からなる $R^p$ の部分集合である．$x, y \in X$ のように個体に添字を付けない場合は，$x = (x^1, x^2, \ldots, x^p)^T$，$y = (y^1, y^2, \ldots, y^p)^T$ のように表す．

次に，2つの個体 $x, y \in X$ 間の非類似度について考える．非類似度とは，個体を2つ与えたとき，その対に対して決まる実数であり，その値が小さければ小さいほど，2つの個体は互いに類似しているとされる．クラスタリングのために用いる非類似度を記号 $d(x, y)$ を用いて表すことにする．

上のように個体が実ベクトル空間の点のとき，その空間に定義された距離（メトリック）の近さによって個体間の類似度が判断されるべきである．最も頻繁に用いられるの

は，ユークリッド距離であり，ノルムの記号を用いれば $x$ と $y$ との距離は

$$\|x-y\| = \sqrt{\sum_{j=1}^{p}(x^j - y^j)^2}$$

で表される．

この距離をクラスタリングのための非類似度として用いる場合，$d(x,y) = \|x-y\|$ となる．また，ユークリッド距離の2乗を非類似度とすることも多い．その場合は

$$d(x,y) = \|x-y\|^2 = \sum_{j=1}^{p}(x^j - y^j)^2 \tag{3.1}$$

となる．

## 3.2　ファジィ集合論

ファジィ集合論は，Zadehにより創始された理論で，物事における曖昧性を境界のはっきりしない集合によって取り扱う方法である．

一般に，集合ある概念を表現するために用いられる．例えば，ある人間の集団における成年男女の集合は，生年月日が不明な人がいない限り，明確に決まる．しかしながら，日常的に使われ，しかも不明確な概念が多く存在する．例えば，ある人が若いかどうか，背が高いかどうかなどは主観的であり，かつ，曖昧さを含んでいる．

ファジィ集合論では $X$ における曖昧な概念を表す'集合のようなもの'をファジィ集合（fuzzy set）と呼ぶ．ファジィ集合 $A$ はメンバーシップ関数 $\mu_A$ で特徴付けられる．$\mu_A$ は任意の $x \in X$ に対して $0 \leq \mu_A(x) \leq 1$ なる実数を連続的に対応させる関数であり，その意味は次のように解釈するものとする．

a. $x \in X$ が $A$ が表す概念に完全にあてはまる場合，$\mu_A(x) = 1$．このとき，$x$ はファジィ集合 $A$ に完全に帰属しているという．

b. $x \in X$ が $A$ が表す概念に完全にあてはまらない場合，$\mu_A(x) = 0$．このとき，$x$ はファジィ集合 $A$ にまったく帰属していないという．

c. $x \in X$ の $A$ が表す概念に対するあてはまり方が曖昧な場合，$0 < \mu_A x < 1$．このとき，$x$ のファジィ集合 $A$ への帰属性は曖昧であるという．

d. $x, y \in X$ について，$y$ の方が $x$ より $A$ の表す概念によくあてはまっていると判断できる場合，$\mu_A(x) < \mu_A(y)$．

要約すると，この解釈は，$x$ に対して $\mu_A(x)$ が ' 要素 $x$ のファジィ集合 $A$ への帰属性の度合い ' であるとみなすことを意味している．

なお，通常の集合（ $B$ とする）に対してその特性関数（$f_B$とする）を対応させると，$f_B(x) = 1(x \in B)$，$f_B(x) = 0(x \notin B)$ である．このように，$f_B$ は上の解釈を満たすので，この集合のメンバーシップ関数と考えてもよい．したがって，クリスプ集合（ファジィでない普通の集合）はファジィ集合の一種であるとみなすことができる．

よく引き合いに出される例として $x$ を人の背の高さとし，$A$ は ' 背が高い ' ことを表すとする．いま，$a < b$ である 2 つのパラメータを与えて

$$\mu_A(x) = \begin{cases} 0 & (x \leq a) \\ \frac{x-a}{b-a} & (a < x \leq b) \\ 1 & (x > b) \end{cases}$$

のようにメンバーシップ関数を折れ線で定めることができる．$a$，$b$ はメンバーシップ関数を定める人によって異なるだろう．ある人は $a = 170(cm)$，$b = 180$ とするかもしれないし，他の人には別の選択があるだろう．このようにファジィ集合論は根本的に主観的なモデリングを許容する考えをもっている．

$\mu_A$ はラベル $A$ をもつ関数にすぎないが，これをファジィ集合と呼ぶのは，$A$ に対して集合の演算や基本的関係を定義するからである．

[a ]（ファジィ集合の包含関係）　ファジィ集合 $A$，$B$ に対して包含関係を次のように定義する．

$$A \subseteq B \Longleftrightarrow \mu_A(x) \leq \mu_B(x) \quad for \quad all \quad x \in X$$

[b ]（ファジィ集合の相等関係）　ファジィ集合 $A$，$B$ に対して包含関係を次のように定義する．

$$A = B \Longleftrightarrow \mu_A(x) = \mu_B(x) \quad for \quad all \quad x \in X$$

[c ]（ファジィ集合の演算）　ファジィ集合 $A$，$B$ に対して合併，共通部分，補集合の各演算を次のように定義する．

$$\begin{aligned} \mu_{A \cup B}(x) &= \max\{\mu_A(x), \mu_B(x)\}, \\ \mu_{A \cap B}(x) &= \min\{\mu_A(x), \mu_B(x)\}, \\ \mu_{\bar{A}}(x) &= 1 - \mu_A(x) \end{aligned}$$

[d ]（ファジィ集合の $\alpha-$ カット）　ファジィ集合 $A$ と与えられた実数 $\alpha(0 \leq \alpha \leq 1)$ に対して，$\alpha-$ カット $A_\alpha$ を $\mu_A(x) \geq \alpha$ を満たす $x \in X$ のクリスプ集合によって定義する．すなわち

$$A_\alpha = \{x \in X : \mu_A(x) \geq \alpha\}$$

なお，$0 \leq \alpha \leq 1$ を $\alpha \in [0,1]$ と書く．$\alpha-$ カットについて次の関係が成立することが知られている．

$$\begin{aligned} A \subseteq B &\iff A_\alpha \subseteq B_\alpha \quad for \ \ all \ \ \alpha \in [0,1] \\ A = B &\iff A_\alpha = B_\alpha \quad for \ \ all \ \ \alpha \in [0,1] \\ (A \cup B)_\alpha &= A_\alpha \cup B_\alpha \\ (A \cap B)_\alpha &= A_\alpha \cap B_\alpha \end{aligned}$$

## 3.3　c-平均法

c-平均法（c-means）は，非階層的方法の代表的手法である．この方法では，クラスター，つまりクラスタリングによって得られたクラスの数 (ここでは c) をあらかじめ指定し，個体を c 個のクラスに分割する．また，非類似度はユークリッド距離の 2 乗（3.1）をとり，クラスターの中心と各個体との間の非類似度を分類の基準にする．

c-平均法は単一のアルゴリズムであると解釈すべきではなく，むしろ，ある概念にもとづく方法の集まりと考えられる．この手法は，もともと MacQueen によって $K$-means と呼ばれていた．MacQueen によれば，

> $K$-means の手続きをインフォーマルに述べれば，次のようになる．$K$ 個ランダムに個体を選び，それぞれのグループの代表とする．ほかの個体を一つずつ選び，最も近い平均値をもつグループに割り当てる．割り当てられたグループについて，平均値を更新する．各ステージにおいて，$K$ 個の平均値がグループを代表する（したがって $K$-means と呼ぶ）．

MacQueen はこれを $K$-means プロセスと呼び，$K$-means によるクラスタリングとは区別している．実際，MacQueen が $K$-means クラスタリングのために開発したプログラムには，クラスター数を更新するパラメータなども含まれていて，以下に述べる標準的アルゴリズムよりも複雑な手続きとなっている．

また，ここでは $K$-means を c-平均法といいかえることにする（以下の注意参照）．このように，クラスタリングすべき対象を最も近いクラスター中心に割り当て，中心を更新する方法は c-平均法ととらえることができる．

最も単純な c-平均法のアルゴリズムを次に述べる．これは Anderberg において Forgy のアルゴリズムと述べられている．なお，**HCM** は Hard c-Means の略であり，ハード（Hard）とはファジィ概念を用いない通常のクラスタリングを指し，ファジィが柔らかい方法であるのに対して通常のクラスタリング技法は固い方法である，という意味合いで使われている．

⟨ アルゴリズム **HCM** ⟩

**HCM1.** [初期値]c 個のクラスター中心あるいは初期分割をランダムに与える．

**HCM2.** [割り当て] 各対象を最も近いクラスター中心に割り当てる．

**HCM3.** [中心の更新] すべての対象の割り当てが一つ前のステップと変化がなければ終了．そうでなければ各クラスターの重心を新しい中心として **HCM** にもどる．

**End of HCM.**

**注意.**　c-平均法の'平均'は，クラスターの中心の各成分がそれに対応する個体データの成分の平均値として計算されることからきている．クリスプの（ファジィでない通常の）クラスタリングを扱う多くの文献において，c-平均法は $K$-means あるいは $K$-平均法として言及されている．c と $K$ との違いは，クラスターの数を c 個と仮定するか，あるいは $K$ 個と仮定するかの違いだけであって，本質的なものではない．ここでは，c-平均法と呼ぶが，$K$-平均法，または原語のまま $K$-means と呼んでもまったく差し支えない．□

次に，ごく単純な数値例を以下に示す．説明のための数値例には，個体の数が 10 個以下のものを用いるが，実際的な問題でこのような少数の個体をクラスタリングすることはまずないので，この種の例は純粋にクラスタリング手法を例示するためのものである．

**例 3.1.** 図 3.1 には 5 つの個体 $x_1, \ldots, x_5$ が平面上に示されている．アルゴリズム **HCM** をこれに適用し，$c = 2$ すなわち 2 つのクラスターを生成する．**HCM1** において初期クラスターを $G_1 = \{x_1, x_2, x_3\}$，$G_2 = \{x_4, x_5\}$ とする．このとき，それぞれのクラスターの重心 $v_1$，$v_2$ は小円 ∘ で表される．

**HCM2** では，$x_3$ は $v_1$ よりも $v_2$ に近いため，$G_1$ から $G_2$ へ移動する．そのほかの個体に

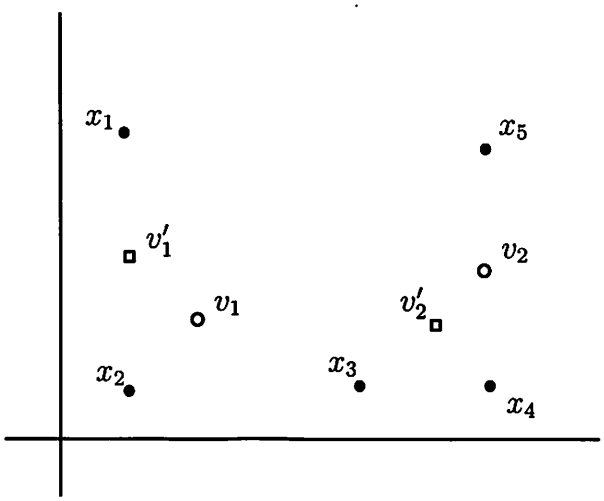


図 3.1: c-平均法の説明のための例

ついてはクラスター間の移動は起こらない．したがって，$G_1 = \{x_1, x_2\}$，$G_2 = \{x_3, x_4, x_5\}$ となり，**HCM3** では重心が更新される．新たな重心 $v_1'$，$v_2'$ は小四角形で表される．

再び **HCM2** にもどると，すべての個体はそのクラスター中心に最も近いため，クラスターの変化はない．したがって，アルゴリズムは終了し，クラスターとして $G_1 = \{x_1, x_2\}$ と $G_2 = \{x_3, x_4, x_5\}$ が得られる．

注意すべきことは，初期値の選び方によっては，適切なクラスターが得られないことがあるということである．上の例で，仮にクラスターを $G_1' = \{x_1, x_5\}$，$G_2' = \{x_2, x_3, x_4\}$ と選んだとすると容易にわかることであるが，クラスターの変化はなく，これらがそのまま最終結果となる．$G_1'$，$G_2'$ が先の $G_1$，$G_2$ に比べて不適切であるのはいうまでもない．□

アルゴリズム **HCM** は単純であるが，さまざまな考察の出発点になる．まず，**HCM2** において最も近いクラスター中心への割り当てという分類規則が用いられることに注意する．これは，教師付き分類 (supervised classification) すなわち外的基準のもとで分類をおこなう方法の一種である最近隣分類規則に対応している．つまり，各クラスターの中心をそのクラスターの代表として，最近隣法の分類規則を適用したものにほかならない．

いま，クラスターの中心が代表点となる場合の最近隣分類規則（最近中心分類）について幾何学的に考察する．簡単のため，個体が属する空間を 2 次元平面とする．図 3.2 に示された点がクラスターの中心であるとする．最近中心分類規則によってあるクラスターに分類される平面上の点の集合は，この図におけるほかの点よりもそのクラスター中心に最も近い点からなるので，図 3.2 における折れ線で囲まれ，対応するクラスター中心を含

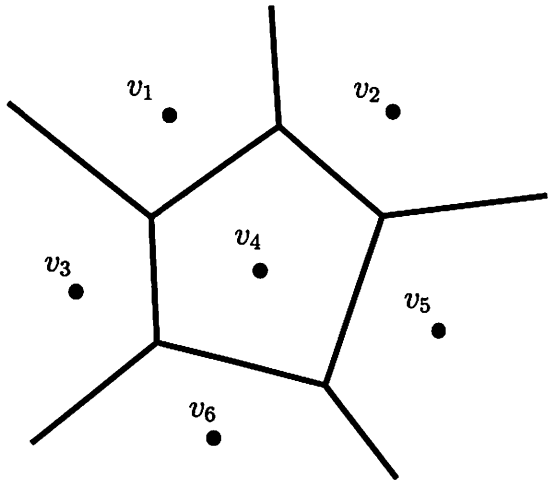


図 3.2: 最近中心分類規則とボロノイ図

む各領域になる．この領域分布は，各点を結ぶ垂直2等分線から構成されるモザイク状の図で，ボロノイ図と呼ばれる．なお，ボロノイ図は，計算幾何学において考察される基本的な主題である．

次にアルゴリズム HCM が最適化に関連していることをみる．

HCM2 における最近中心分類は，

$$x_k \in G_l \Longleftrightarrow v_l = arg\min_q \|x_k - v_q\|^2 \tag{3.2}$$

の形の最適化であり，$\|x_k - v\|^2$ が最小となるクラスターに $x_k$ を割り当てている．

一方，HCM3 における重心 $M(G)$ は次の最適解である．

$$M(G) = arg\min_v \sum_{s\in G} \|x - v\|^2 \tag{3.3}$$

なぜなら，

$$F = \sum_{x\in G} \|x - v\|^2$$

とおくとき，$F$ は成分 $v^j, j = 1, \ldots, p$ の2次関数であり，

$$\frac{1}{2}\frac{\partial F}{\partial v^j} = |G|v^j - \sum_{x\in G} x^j = 0$$

から（3.3）が得られるからである．

これらのことから，HCM2，HCM3では次の最適化を行っていることがわかる．

**HCM2.** 与えられた $v_1,\ldots,v_c$ に対して，

$$j=\sum_{i=1}^{c}\sum_{x\in G_i}\|x-v_i\|^2 \tag{3.4}$$

が最小になるようにクラスター $g=\{G_1,\ldots,G_c\}$ を定める．

**HCM3.** 同じ（3.4）において，$G_1,\ldots,G_c$ が与えられたとみなし，$v_1,\ldots,v_c$ を変化させて最小化する．

つまり，同じ目的関数を別の変数によって繰り返し最適化しているのである．

同一の目的関数をこのように繰り返し最適化しているため（3.4）の値は単調に減少していく．また，HCM2におけるクラスターの選び方は有限個しかなく，クラスターが変化するときには，目的関数の値は同じではない．よって次のことがわかる．

**命題2.1.**　アルゴリズムHCMは有限回の繰り返しで終了する．

しかしながら，このことはこのアルゴリズムの効率がよいことを必ずしも意味しない．実際には，HCM2における'すべてのクラスターの選び方'は組み合わせの数が膨大であるからである．

## 3.4　標準ファジィc-平均法

ファジィクラスタリングは，個体がクラスに帰属する度合いに曖昧さを認めるという考えに基づいている．曖昧さの表現はファジィ理論による要素の集合への帰属度によって表される．

いま，個体 $x_k$ がクラス $G_i$ に帰属するということを変数 $u_{ik}=1$ と表し，個体 $x_k$ がクラス $G_i$ に属さないということを $u_{ik}=0$ と書くとする．これは，属するか属さないかの2値的判断を2値変数 $u_{ik}$ によって表している．集合の記号を用いると，

$$x_k\in G_i \Longleftrightarrow u_{ik}=1$$

$$x_k\notin G_i \Longleftrightarrow u_{ik}=0$$

であり, $u_{ik}$ は集合の特性関数に一致している.

これに対し, ファジィ集合は特性関数の連続値への一般化による集合概念の拡張である. そこで, 上の変数 $u_{ik}$ に0から1までの区間の値を連続的にとることを許し, その値の個体 $x_k$ のクラス $G_i$ への帰属性の度合いとみなすことにする. 実際, 様々なファジィクラスタリング技法において, この方法でファジィネス（fuzzyness:ファジィである性質のこと）が導入されてきている. このように, ファジィ理論の基本的考え方は, 曖昧さを0から1までの区間の値で表すという点にある.

ファジィクラスタリングにおける多くの研究はファジィc-平均法に集中している. その理由は, この方法が理論的に整理されていて最も見通しが良く, 様々な変形がなされていて, 多くの応用が試みられ, 有用性が実証されているからである. そこでは主にファジィ平均法について述べる.

まず, 先に述べたアルゴリズム HCM を2値変数を用いた最適化問題の形に書いてみる. クラスタリングのための個体は $p$ 次元ユークリッド空間の点 $x_k = (x_k^1, \ldots, x_k^p)^T, k = 1, \ldots, n$ で表される. 変数を $c \times n$ 行列として $U = (u_{ik})$ の形に書く. また, クラスター中心 $v_i = (v_i^1, \ldots, v_i^p)_T$ をまとめて $V = (v_1, \ldots, v_c)$ と表しておく. なお, 先のc-平均法の場合と違って, $v_i$ は必ずしもクラスターの重心ではない.

そこで, 次の目的関数を考える.

$$J_{hcm}(U, V) = \sum_{k=1}^{n} \sum_{i=1}^{c} u_{ik} \|x_k - vi\|^2 \tag{3.5}$$

ここで, 先に述べたように, $\|x_k - v_i\|$ はユークリッド距離を表すノルムの記号である. なお, 以下に現れる目的関数の添字は, その目的関数に対応する技法を表している. 例えば, $J_{hcm}$ はこの関数が HCM に対応しているということを表している.

一方, 最適化における制約を

$$M_c = \{(u_{ik}) \ : \ u_{ik} \in \{0, 1\}, \sum_{i=1}^{c} u_{ik} = 1 \ for \ all \ k\}$$

と表す. この制約は $u_{ik}$ の値が0または1に限られ, すべての個体 $x_k$ についてどれか1つの $i$ に対応する $u_{ik}$ だけが1であり, ほかは0であることを意味している. したがって, $M_c$ の条件を満たす $u_{ik}$ によって $x_k$ がクラスターに分けられる.

そこで, 次のアルゴリズム HCM" を考える.

〈アルゴリズム HCM"〉

**HCM"1.** $\bar{U}$ と $\bar{V}$ の初期値を定める.

**HCM"2.** $\bar{V}$ を固定して,

$$\min_{U \in M_c} J_{hcm}(U, \bar{V})$$

を解き,最適解を $\bar{U}$ とする.

**HCM"3.** $\bar{U}$ を固定して,

$$\min_{V} J_{hcm}(\bar{U}, V)$$

を解き,最適解を $\bar{V}$ とする( $V$ は制約されていないことに注意).

**HCM"4.** 解 $(\bar{U}, \bar{V})$ が収束すれば終了.そうでなければ **HCM"2** へ.

**End of HCM"**

アルゴリズム HCM" が HCM と等価であることは,HCM が最適化を行っていることを論じた部分をみればわかる.実際,$\bar{V} = (\bar{v}_1, \ldots, \bar{v}_c)$ を固定したとき,HCM"2 における最適解は

$$\bar{u}_{ik} = 1 \Longleftrightarrow \bar{v}_i = arg \min_l \|x_k - \bar{v}_l\| \tag{3.6}$$

$$\bar{u}_{ik} = 0, \quad j \neq i \tag{3.7}$$

となる.また,$\bar{U}$ を固定するとき,HCM"3 における最適解は

$$\bar{v}_i = \frac{\sum_{k=1}^{n} \bar{u}_{ik} x_k}{\sum_{k=1}^{n} \bar{u}_{ik}}$$

で与えられる.先に述べたように,

$$G_i = \{x_k \in X : \bar{u}_{ik} = 1\}$$

とすれば, $\bar{U}$ の式は最も近い中心への割り当てであり, $\bar{v}_i$ は $G_i$ の重心である.

HCM" の形の交互最適化は以下の論議において繰り返し現れる.実際,ファジィc-平均法は上の形の交互最適化において $J_{hcm}$ とは異なる目的関数とファジィされた制約条件を仮定することによって生まれる.

いま，HCM" において，行列 $U=(u_{ik})$ の要素が必ずしも2値ではなく，0から1の間の値をとることを許すとする．ただし，各個体についてすべてのクラスターに対する帰属度を加えたものが1になるという正規化の条件を付け加えておくとする．このように曖昧なクラスターに分けることをファジィ分割（fuzzy partition）という．このとき，制約 $M_c$ を緩め，

$$M_f=\{(u_{ik})\ :\ u_{ik}\in[0,1],\sum_{i=1}^{c}u_{ik}=1\ for\ all\ k\}$$

と $u_{ik}$ に実数値を許容することによってファジィ分割が表現される．

ここで，HCM"2における最適解は一般にスクリプト解となる．なぜなら，$J_{hcm}(U,V)=\sum_{k=1}^{n}\sum_{i=1}^{c}u_{ik}\|x_k-v_i\|^2$ は $u_{ik}$ について線形であり，制約 $M_f$ も線形であるので，この問題は線形計画となり，最適解は制約条件が形成する多角形の端点であるとして充分である．ところが，端点は明らかにクリスプな（ファジィでない通常の）解(3.6)になるからである．したがって，制約をこのように緩和してもファジィネスを導入したことにはならない．

ファジィクラスタリングの方法に導くためには，目的関数を変更する必要があると考え，DunnとBezdekは次の形の目的関数を導入した．

$$J_{fcm}(U,V)=\sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})^m\|x_k-v_i\|^2=\sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})^m d_{ik}\qquad(3.8)$$

ここで，$m$ は $m>1$ を満たすように選ばれるべきパラメータである（$m=1$ とすると前の目的関数になる.）また，記号を簡単にするため，$d_{ik}=\|x_k-v_i\|^2$ とおいている．

標準的なc-平均法のアルゴリズムは，HCM" において $J_{hcm}$ の代わりに $J_{fcm}$ を，$M_c$ の代わりに $M_f$ を用いたものである．このアルゴリズム FCM を次に述べる．

〈アルゴリズム FCM(Fuzzy c-Means)〉

**FCM0.** （仮定）　$J(U,V) = J_{fcm}(U,V)$ とする．

**FCM1.** $\bar{V}$ の初期値を定める．

**FCM2.** $\bar{V}$ を固定して，

$$\min_{U \in M_f} J(U, \bar{V})$$

を解き，最適解を $\bar{U}$ とする．

**FCM3.** $\bar{U}$ を固定して，

$$\min_{V} J(\bar{U}, V)$$

を解き，最適解を $\bar{V}$ とする（ $V$ は制約されていないことに注意）．

**FCM4.** 解 $(\bar{U}, \bar{V})$ が収束すれば終了．そうでなければ $\hat{U} = \bar{U}$ ，　$\hat{V} = \bar{V}$ として FCM2 へ．

**End of FCM**

なお，FCM4 の収束条件については，次の（i），(ii)，(iii）のいずれかが用いられる．

(i)　小さな正数 $\epsilon > 0$ を与え，$\bar{U}$ と一つ前の最適解 $\hat{U}$ との差が

$$\max_{i,k} |\bar{u}_{ik} - \hat{u}_{ik}| < \epsilon$$

を満たすとき収束したと判定する．

(ii)　小さな正数 $\epsilon > 0$ を与え，$\bar{V}$ と一つ前の最適解 $\hat{V}$ との差が

$$\max_{1 \le i \le c} \|\bar{v}_i - \hat{u}_i\| < \epsilon$$

を満たすとき収束したと判定する．

(iii)　目的関数が減少しなくなったとき収束したと判定する．

(iv)　このほかに最大繰り返し数を設定しておく．

**FCM2** における最適解は次の式で与えられる．まず，すべての $v_i, i = 1, \ldots, c$ に対して $x_k \neq v_i$ であるような $x_k$ については

$$u_{ik} = \left[ \sum_{j=1}^{c} \left( \frac{\|x_k - \bar{v}_i\|^2}{\|x_k - \bar{v}_j\|^2} \right)^{\frac{1}{m-1}} \right]^{-1}, \quad for\ x_k \neq v_i, i = 1, \ldots, c \tag{3.9}$$

ある $v_i$ について，$x_k = v_i$ となる $x_k$ については，

$$u_{ik} = 1; \quad u_{jk} = 0\ (j \neq i). \tag{3.10}$$

**FCM3** における最適解は，

$$v_i = \frac{\sum_{k=1}^{n} (\bar{u}_{ik})^m x_k}{\sum_{k=1}^{n} (\bar{u}_{ik})^m} \tag{3.11}$$

このように，個体と一致する中心がある場合を除いて，解（3.9）はファジィ解，すなわち0と1の間の値をとる．

これらの式で最適解が与えられることを説明する．まず，**FCM2** における $U$ の最適解について考える．いま，$M_f$ をさらに緩めて，制約条件を $\sum_{i=1}^{c} u_{ik} = 1$ のみとし，

$$u_{ik} \geq 0, \quad 1 \leq i \leq c, \quad 1 \leq k \leq n \tag{3.12}$$

の条件を取り去ってみる．(3.12) なしで得られた最適解が仮に $M_f$ に入っていたとすると，緩い条件のもとでの最適解であるのだから，制約 $M_f$ のもとでの最適解でもあるはずである．$J_{fcm}$ を $\sum_{i=1}^{c} u_{ik} = 1$ のもとで $U$ に関して最小にする問題はラグランジュ乗数法を用いて解ける．

ラグランジュ乗数を $\lambda_k, k = 1, \ldots, n$，ラグランジュ関数を

$$L = J_{fcm} + \sum_{k=1}^{n} \lambda_k \left( \sum_{i=1}^{c} u_{ik} - 1 \right)$$

とすると，最適性の必要条件として

$$\frac{\partial L}{\partial u_{ik}} = m(u_{ik})^{m-1} d_{ik} + \lambda_k = 0$$

が得られる．$x_k$ について $x_k = v_i$ となる $v_i$ が存在しないとする．このとき，$d_{jk} > 0, j = 1, \ldots, c$ であることがわかる．$\lambda_k$ を消去するため，上式の添字 $i$ を $j$ に変更して移項すれば，

$$u_{jk} = \left[ \frac{-\lambda_k}{m d_{jk}} \right]^{\frac{1}{m-1}} \tag{3.13}$$

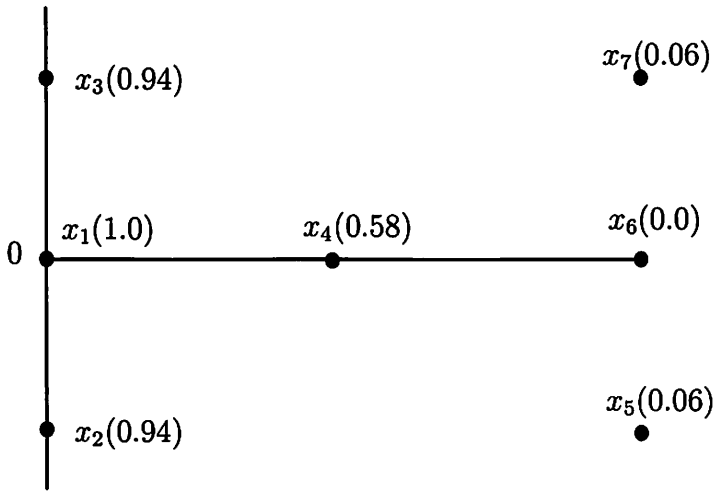


図 3.3: 平面上の 7 つの個体と 2 つのファジィクラスター

この両辺を $j=1,\ldots,c$ について加えると，$\sum_{j=1}^{c} u_{ik}=1$ より，

$$\sum_{j=1}^{c}\left[\frac{-\lambda_k}{md_{jk}}\right]^{\frac{1}{m-1}}=1$$

この式と（3.13）から $\lambda_k$ を消去することができて，

$$u_{ik}=\left[\sum_{j=1}^{c}\left(\frac{d_{ik}}{d_{jk}}\right)^{\frac{1}{m-1}}\right]^{-1}$$

を得る．この解は $u_{ik}>0$ を満たしているから先に述べた注意によって **FCM2** の最適解である．なお，最適解の必用条件だけからは，一般に最適解ということはできないが，この場合，目的関数が $U$ に関して凸であるため，実際に最適解であることが分かる．なお，$x_k=v_i$ を満たす $v_i$ が存在する場合，$u_{ik}=1$ が最適であることは自明である．

**FCM3** については，制約がないため，$J_{fcm}$ を $v_i^j$ について偏微分すれば，(3.11) が得られる．

**例 3.1.**　図 3.3 には 7 個の個体（$n=7$）が平面上に点として示されている．これらを 2 つ（$c=2$）のファジィクラスターに分けた結果の帰属度（メンバーシップ）が括弧の中に示されている．クラスターが 2 つであるため，1 つのクラスターに対するメンバーシップだけを示せば，他方のクラスターに対するメンバーシップは 1 からこれらの値を減じることによって $u_{2k}=1-u_{1k}, k=1,\ldots,7$ と計算される．□

## 3.5　k-最近傍法による語義判別

k-最近傍法は外的基準のもとで分類を行う方法である．この方法は，すでにクラス分けされた有限個の個体 $x_1,\ldots,x_n$ の集合が仮定され，新たな個体データ $y$ が得られたとき，これをどのクラスに割り当てるかの規則を定める．これらのクラスをクラスターと同じ記号 $G_1,\ldots,G_c$ で表す．

最近傍法では，$y$ に最も近い個体が属するクラスに $y$ を割り当てる．式で書くと，

$$x_l = arg\min_{x\in X}\|x-y\|$$

となる．$x_l$ が $G_q$ の要素（$x_l \in G_q$）であるときに，$y$ を $G_q$ に割り当てる．

それを拡張したk-最近傍法では，$y$ に最も近いものから順に $k$ 個の個体をとり，これら $k$ 個の個体のうち，個体の数が最も多いクラスを選ぶ[4]．たとえば，$k=7$ として，$y$ に近いものから順に7つとったとき，その中に $G_1$ からの個体が1つ，$G_2$ からの個体が4つ，$G_3$ からの個体が2つであれば，$y$ は $G_2$ に属すると定められる．

ここでは，ラベル付き訓練データを各クラスに設定し，ラベルなし訓練データをどのクラスに割り当てるかを判別した．また，ここでの $k$ は5に設定した．

## 3.6　ファジィクラスタリングによる教師なし学習

まずラベル付きの各訓練事例を1つのクラスに対応させる．その事例自身がクラスタであり，しかもクラスのプロトタイプともなっている．次にラベルなしの訓練事例をそれらのクラスタにクラスタリングさせる．クラスタリングによって最初に設定してあるプロトタイプの移動が起こる．移動した先のプロトタイプをラベル付きの訓練事例と見て，k-最近傍法によって識別を行う．

なお，ここでFCMを本手法で用いる際にラベル付き事例の扱いには注意が必要である．FCM1回目のループのFCM2のステップでは，一般にラベル付きの事例はプロトタイプと一致していない．しかしこの場合でもラベル付きの事例はクラスが確定しているので（3.9）ではなく，(3.10）を用いる．

## 3.7　利用した素性集合

ファジィクラスタリングを行うためには，各事例を $p$ 次のベクトルで表現すればよい．一般に分類問題の事例は素性の集合で表されるので，各素性を次元に割り当て，その素

性が存在する場合にその次元の値を 1 に設定し，存在しない場合に 0 に設定することで，各事例が $p$ 次のベクトルで表現できる．

ここではで利用した素性について述べる．

まず語義の曖昧性解消の手がかりとなる属性として以下のものを設定した．

| | |
|---|---|
| e1 | 直前の単語 |
| e2 | 直後の単語 |
| e3 | 前方の内容語２つまで |
| e4 | 後方の内容語２つまで |
| e5 | e3 の分類語彙表の番号 |
| e6 | e5 の分類語彙表の番号 |

例えば，語義判別対象の単語を「出す」として，以下の文を考える（形態素解析され各単語は原型に戻されているとする）．

短い/コメント/を/出す/に/とどまる/た/。

この場合,「出す」の直前，直後の単語は「を」と「に」なので，‘e1=を’，‘e2=に’となる．次に,「出す」の前方の内容語は「短い」と「コメント」なので，‘e3=短い’，‘e3=コメント’の２つが作られる．またここでは句読点も内容語に設定しているので,「出す」の後方の内容語は「とどまる」「。」となり，‘e4=とどまる’，‘e4=。’が作られる．次に「短い」の分類語彙表 [10] の番号を調べると， 3.1920_1 である．ここでは分類語彙表の 4 桁目と 5 桁目までの数値をとることにした．つまり‘e3=短い’に対しては，‘e5=3192’ と ‘e5=31920’ が作られる.「コメント」は分類語彙表には記載されていないので，‘e3=コメント’に対しては e5 に関する素性は作られない．次は「とどまる」の分類語彙表を調べるはずだが，ここでは平仮名だけで構成される単語の場合，分類語彙表の番号を調べないことにしている．これは平仮名だけで構成される単語は多義性が高く，無意味な素性が増えるので，その問題を避けたためである．もしも分類語彙表上で多義になっていた場合には，それぞれの番号に対して並列にすべての素性を作成する．

結果として，上記の例文に対しては以下の８つの素性が得られる．

e1=を，　e2=に，　e3=短い，　e3=コメント，

e4=とどまる，　e4=。，　e5=3192，　e5=31920，

上記の例のようにして，「出す」に対するすべての訓練事例の素性を集め，各素性に 1 番から順に番号をつける．例えば，本論文の実験では「出す」に対しては 978 種類の素性があり，$p = 978$ となる．また上記例の素性には表 3.1 のように番号が振られた．

表 3.1: 素性と次元番号

| 素性 | 次元番号 |
|---|---|
| e1=を | 21 |
| e2=に | 60 |
| e3=コメント | 134 |
| e3=短い | 302 |
| e4=。 | 379 |
| e4=とどまる | 406 |
| e5=3192 | 789 |
| e5=31920 | 790 |

以上より，上記例文に対するベクトルは第 21 次元目，第 60 次元目，第 134 次元目，第 302 次元目，第 379 次元目，第 406 次元目，第 789 次元目，第 790 次元目の各要素が 1 であり，その他の要素がすべて 0 の 978 次元のベクトルとなる．

# 第4章　実験

## 4.1　語義判別問題

本手法の有効性を確認するために，SENSEVAL2の日本語辞書タスクで課題とされた名詞50単語に関する語義判別を試みた．

SENSEVAL2の日本語辞書タスクは，単純な語義判別問題である．対象単語は名詞50単語，動詞50単語の計100単語である．これら100単語は語義の頻度分布のエントロピーを考慮して選定されており，語義判別が容易なものから困難なものまでバランス良く選定されている．ラベル付きの訓練事例は1単語平均して名詞は177.4事例，動詞は172.7事例用意されている．またテストデータは各単語に対して100問のテストが用意されている．つまり名詞に対しては計5000問，動詞に対しても計5000問のテストが行える．ただし，ラベルなし訓練事例はSENSEVAL2からは提供されていない．これには通常のテキストが使えそうだが，実際は制限がある．それはラベル付き訓練事例を作成した際に用いた単語辞書や品詞分類を合わせる必要があるからである．そのためここではRWCテキストデータベース第2版に納められた毎日新聞95年度版の1年分の記事を利用して，ラベルなし訓練事例を収集した．このデータはラベル付き訓練事例のもとになったデータであり，同一の形態素解析システムを用いて形態素解析されている．収集できたラベルなし訓練事例の数は1単語平均して名詞は7571.2事例，動詞は6571.9事例である．

各単語の事例数のリストを表4.1および表4.2に示す．

## 4.2　Naive Bayes + EMアルゴリズム

名詞50単語に対して，Naive Bayes（表中のNB），収束するまで繰り返し実行したEMアルゴリズム（表中のEM1,...,EM10）の各結果を表4.3に示す．ラベル付き訓練データ

表 4.1: 各単語の事例数（名詞）

| 単語 | ラベル付き訓練事例 | ラベルなし訓練事例 |
|---|---|---|
| aida | 134 | 763 |
| atama | 69 | 2980 |
| ippan | 167 | 5089 |
| ippou | 174 | 8803 |
| ima | 229 | 11671 |
| imi | 73 | 3624 |
| utagai | 101 | 3043 |
| otoko | 113 | 676 |
| kaihatsu | 109 | 6697 |
| kaku_n | 155 | 4419 |
| kankei | 314 | 10080 |
| kimochi | 156 | 3911 |
| kiroku | 136 | 4702 |
| gijutsu | 98 | 5042 |
| genzai | 241 | 20936 |
| koushou | 142 | 4832 |
| kokunai | 177 | 5434 |
| kotoba | 163 | 5065 |
| kodomo | 254 | 8774 |
| gogo | 296 | 11805 |
| shijo | 153 | 5771 |
| shimin | 107 | 5796 |
| shakai | 239 | 9140 |
| shonen | 90 | 2052 |
| jikan | 183 | 12590 |
| jigyou | 153 | 7287 |
| jidai | 260 | 9267 |
| jibun | 262 | 10001 |
| joho | 185 | 9858 |
| sugata | 101 | 4697 |
| seishin | 57 | 2408 |
| taishou | 136 | 5908 |
| daihyou | 366 | 11738 |
| chikaku | 138 | 5572 |
| chihou | 171 | 4638 |
| chushin | 155 | 7695 |
| te | 127 | 6851 |
| teido | 102 | 4047 |
| denwa | 164 | 9145 |
| doujitsu | 134 | 3713 |
| hana | 75 | 3371 |
| hantai | 141 | 4837 |
| baai | 192 | 7869 |
| mae | 326 | 24648 |
| minkan | 74 | 3131 |
| musume | 102 | 2198 |
| mune | 56 | 1649 |
| me | 128 | 16649 |
| mono | 654 | 26037 |
| mondai | 536 | 21651 |
| 平均 | 177.4 | 7571.2 |

表 4.2: 各単語の事例数（動詞）

| 単語 | ラベル付き訓練事例 | ラベルなし訓練事例 |
|---|---|---|
| ataeru | 116 | 3909 |
| iu | 366 | 16426 |
| ukeru | 357 | 13599 |
| uttaeru | 70 | 3599 |
| umareru | 65 | 2907 |
| egaku | 70 | 3192 |
| omou | 465 | 20229 |
| kau | 87 | 2311 |
| kakaru | 115 | 4783 |
| kaku_v | 135 | 5634 |
| kawaru | 97 | 4029 |
| kangaeru | 291 | 12852 |
| kiku | 180 | 7031 |
| kimaru | 117 | 3552 |
| kimeru | 228 | 8371 |
| kuru | 128 | 5651 |
| kuwaeru | 109 | 3507 |
| koeru | 119 | 3908 |
| shiru | 246 | 7600 |
| susumu | 112 | 4699 |
| susumeru | 132 | 5446 |
| dasu | 208 | 9486 |
| chigau | 105 | 3847 |
| tsukau | 270 | 9088 |
| tsukuru | 183 | 6504 |
| tsutaeru | 97 | 3910 |
| dekiru | 75 | 2869 |
| deru | 412 | 15890 |
| tou | 71 | 3341 |
| toru | 112 | 5705 |
| nerau | 67 | 2318 |
| nokosu | 98 | 2902 |
| noru | 64 | 2450 |
| hairu | 278 | 10541 |
| hakaru | 84 | 2892 |
| hanasu | 175 | 8442 |
| hiraku | 224 | 6867 |
| fukumu | 92 | 3832 |
| matsu | 83 | 2088 |
| matomeru | 93 | 3855 |
| mamoru | 93 | 2858 |
| miseru | 100 | 3895 |
| mitomeru | 212 | 6770 |
| miru | 408 | 14446 |
| mukaeru | 93 | 3383 |
| motsu | 267 | 11325 |
| motomeru | 306 | 11152 |
| yomu | 106 | 2507 |
| yoru | 474 | 13613 |
| wakaru | 178 | 8583 |
| 平均 | 172.7 | 6571.9 |

のみから学習するNaive Bayesの正解率が76.78％であった．EMアルゴリズムを適用すると，1回目の正解率は76.78％となるが，回を重ねるごとに正解率は減少し，最終的にはNaive Bayesの正解率を下回ってしまった．

同様に動詞50単語に対して，得られた結果が表4.4である．動詞による実験ではNaive Bayesによる正解率が78.16％であったが，EMアルゴリズムを用いることで，1回目の正解率は78.44％となり，正解率が向上している．さらに，EMアルゴリズムを収束まで繰り返すことによって最終的には78.70％となり1回目よりも正解率が向上した．

表 4.3: 実験結果（EM：名詞）

| 単語 | NB | EM1 | EM2 | EM3 | EM4 | EM5 | EM6 | EM7 | EM8 | EM9 | EM10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| aida | 81 | 82 | 80 | 79 | 80 | 80 | 80 | 80 | 80 | 80 | 80 |
| atama | 60 | 65 | 65 | 65 | 66 | 64 | 63 | 63 | 63 | 63 | 63 |
| ippan | 88 | 89 | 88 | 87 | 87 | 89 | 86 | 87 | 87 | 86 | 85 |
| ippou | 82 | 87 | 89 | 88 | 88 | 88 | 88 | 88 | 87 | 87 | 88 |
| ima | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 |
| imi | 45 | 51 | 50 | 50 | 50 | 52 | 53 | 53 | 53 | 53 | 54 |
| utagai | 100 | 100 | 100 | 98 | 97 | 97 | 96 | 96 | 95 | 95 | 95 |
| otoko | 92 | 90 | 92 | 92 | 90 | 90 | 89 | 88 | 89 | 89 | 89 |
| kaihatsu | 62 | 62 | 63 | 63 | 63 | 63 | 64 | 64 | 64 | 64 | 64 |
| kaku_n | 71 | 72 | 74 | 76 | 78 | 80 | 80 | 81 | 80 | 77 | 77 |
| kankei | 85 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 |
| kimochi | 65 | 66 | 64 | 66 | 66 | 65 | 65 | 64 | 64 | 64 | 64 |
| kiroku | 74 | 77 | 73 | 73 | 72 | 71 | 71 | 71 | 71 | 71 | 70 |
| gijutsu | 96 | 96 | 95 | 94 | 94 | 92 | 92 | 92 | 92 | 91 | 90 |
| genzai | 97 | 98 | 91 | 87 | 85 | 68 | 42 | 18 | 12 | 9 | 9 |
| koushou | 100 | 100 | 100 | 100 | 100 | 96 | 93 | 93 | 92 | 92 | 88 |
| kokunai | 46 | 48 | 49 | 54 | 54 | 57 | 58 | 58 | 58 | 57 | 58 |
| kotoba | 45 | 41 | 41 | 40 | 41 | 40 | 40 | 40 | 41 | 41 | 41 |
| kodomo | 67 | 72 | 73 | 73 | 73 | 71 | 71 | 69 | 69 | 69 | 68 |
| gogo | 77 | 83 | 86 | 85 | 82 | 80 | 78 | 73 | 69 | 66 | 65 |
| shijo | 77 | 72 | 68 | 61 | 57 | 55 | 57 | 57 | 54 | 54 | 55 |
| shimin | 67 | 61 | 59 | 60 | 62 | 63 | 61 | 62 | 62 | 63 | 63 |
| shakai | 82 | 83 | 83 | 83 | 83 | 83 | 83 | 83 | 83 | 83 | 83 |
| shonen | 92 | 91 | 91 | 91 | 91 | 91 | 90 | 90 | 90 | 90 | 90 |
| jikan | 54 | 54 | 53 | 51 | 50 | 49 | 45 | 43 | 40 | 27 | 15 |
| jigyou | 69 | 68 | 68 | 70 | 71 | 70 | 69 | 69 | 69 | 69 | 70 |
| jidai | 72 | 77 | 78 | 76 | 77 | 76 | 76 | 77 | 77 | 78 | 78 |
| jibun | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| joho | 77 | 70 | 66 | 66 | 66 | 64 | 61 | 60 | 61 | 62 | 61 |
| sugata | 55 | 61 | 61 | 59 | 61 | 61 | 63 | 63 | 63 | 62 | 62 |
| seishin | 65 | 65 | 65 | 66 | 66 | 66 | 66 | 66 | 67 | 66 | 66 |
| taishou | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 |
| daihyou | 85 | 87 | 88 | 87 | 90 | 96 | 98 | 97 | 96 | 95 | 95 |
| chikaku | 74 | 79 | 84 | 86 | 86 | 87 | 88 | 89 | 89 | 89 | 89 |
| chihou | 70 | 67 | 68 | 69 | 72 | 72 | 72 | 73 | 72 | 72 | 69 |
| chushin | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 |
| te | 47 | 46 | 46 | 47 | 47 | 47 | 48 | 48 | 48 | 48 | 48 |
| teido | 100 | 100 | 100 | 99 | 97 | 97 | 96 | 95 | 94 | 93 | 90 |
| denwa | 84 | 83 | 85 | 82 | 80 | 75 | 73 | 71 | 70 | 69 | 65 |
| doujitsu | 81 | 78 | 64 | 57 | 54 | 53 | 54 | 52 | 52 | 52 | 51 |
| hana | 99 | 99 | 99 | 99 | 98 | 97 | 97 | 97 | 97 | 95 | 95 |
| hantai | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 98 | 98 | 98 | 98 |
| baai | 82 | 87 | 92 | 92 | 91 | 90 | 91 | 92 | 91 | 90 | 89 |
| mae | 86 | 88 | 91 | 92 | 91 | 90 | 90 | 88 | 87 | 87 | 85 |
| minkan | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 99 | 99 | 99 |
| musume | 88 | 88 | 88 | 88 | 88 | 88 | 88 | 88 | 88 | 88 | 88 |
| mune | 71 | 79 | 76 | 78 | 77 | 77 | 77 | 76 | 77 | 77 | 77 |
| me | 18 | 18 | 16 | 17 | 17 | 17 | 13 | 13 | 13 | 13 | 13 |
| mono | 31 | 27 | 28 | 28 | 27 | 27 | 27 | 27 | 26 | 26 | 26 |
| mondai | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 |
| 平均 | 76.78 | 77.54 | 77.20 | 76.88 | 76.70 | 76.08 | 75.24 | 74.50 | 74.04 | 73.44 | 72.82 |

表 4.4: 実験結果（EM：動詞）

| 単語 | NB | EM1 | EM2 | EM3 | EM4 | EM5 | EM6 | EM7 | EM8 | EM9 | EM10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ataeru | 71 | 78 | 78 | 78 | 78 | 78 | 78 | 78 | 78 | 78 | 78 |
| iu | 94 | 94 | 94 | 94 | 94 | 94 | 94 | 94 | 94 | 94 | 94 |
| ukeru | 59 | 62 | 64 | 63 | 64 | 63 | 65 | 65 | 64 | 64 | 65 |
| uttaeru | 84 | 85 | 84 | 86 | 86 | 87 | 87 | 88 | 88 | 87 | 87 |
| umareru | 69 | 74 | 81 | 81 | 81 | 82 | 83 | 83 | 83 | 83 | 83 |
| egaku | 58 | 56 | 56 | 53 | 54 | 56 | 56 | 57 | 56 | 56 | 56 |
| omou | 90 | 89 | 89 | 89 | 90 | 91 | 92 | 92 | 92 | 92 | 92 |
| kau | 83 | 83 | 83 | 83 | 83 | 83 | 83 | 83 | 83 | 83 | 83 |
| kakaru | 58 | 55 | 56 | 57 | 57 | 57 | 57 | 57 | 57 | 57 | 57 |
| kaku_v | 72 | 68 | 66 | 67 | 66 | 66 | 65 | 68 | 68 | 68 | 67 |
| kawaru | 92 | 92 | 92 | 92 | 92 | 92 | 92 | 92 | 92 | 92 | 92 |
| kangaeru | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 |
| kiku | 56 | 55 | 55 | 55 | 55 | 55 | 55 | 55 | 55 | 55 | 55 |
| kimaru | 96 | 96 | 96 | 96 | 95 | 94 | 94 | 94 | 94 | 94 | 93 |
| kimeru | 93 | 93 | 93 | 93 | 93 | 93 | 93 | 93 | 93 | 93 | 93 |
| kuru | 84 | 85 | 86 | 85 | 85 | 85 | 85 | 85 | 85 | 86 | 86 |
| kuwaeru | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 89 |
| koeru | 78 | 78 | 80 | 80 | 84 | 88 | 88 | 84 | 85 | 84 | 82 |
| shiru | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 |
| susumu | 49 | 49 | 50 | 48 | 50 | 49 | 50 | 50 | 50 | 50 | 50 |
| susumeru | 97 | 97 | 97 | 97 | 97 | 96 | 96 | 96 | 96 | 95 | 94 |
| dasu | 35 | 31 | 35 | 36 | 35 | 33 | 30 | 30 | 30 | 29 | 29 |
| chigau | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| tsukau | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 |
| tsukuru | 69 | 70 | 74 | 78 | 76 | 75 | 76 | 76 | 76 | 76 | 75 |
| tsutaeru | 75 | 76 | 76 | 75 | 76 | 76 | 76 | 76 | 76 | 75 | 75 |
| dekiru | 81 | 81 | 80 | 80 | 81 | 81 | 81 | 81 | 81 | 81 | 81 |
| deru | 59 | 58 | 60 | 61 | 64 | 64 | 64 | 64 | 64 | 64 | 63 |
| tou | 69 | 73 | 75 | 77 | 78 | 79 | 79 | 78 | 79 | 79 | 79 |
| toru | 32 | 34 | 34 | 35 | 35 | 37 | 37 | 34 | 34 | 34 | 35 |
| nerau | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 99 |
| nokosu | 79 | 79 | 79 | 79 | 79 | 79 | 79 | 79 | 79 | 79 | 79 |
| noru | 54 | 54 | 54 | 54 | 54 | 54 | 54 | 54 | 54 | 54 | 54 |
| hairu | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 | 36 |
| hakaru | 92 | 92 | 92 | 92 | 92 | 92 | 92 | 92 | 92 | 92 | 92 |
| hanasu | 100 | 100 | 100 | 100 | 100 | 97 | 93 | 90 | 89 | 88 | 87 |
| hiraku | 86 | 87 | 89 | 91 | 92 | 93 | 94 | 94 | 94 | 94 | 95 |
| fukumu | 99 | 99 | 99 | 99 | 99 | 99 | 99 | 98 | 98 | 98 | 98 |
| matsu | 52 | 51 | 49 | 51 | 50 | 50 | 50 | 50 | 48 | 48 | 49 |
| matomeru | 79 | 79 | 80 | 80 | 80 | 80 | 80 | 80 | 80 | 80 | 80 |
| mamoru | 79 | 79 | 77 | 77 | 75 | 73 | 73 | 74 | 71 | 71 | 70 |
| miseru | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 | 98 |
| mitomeru | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 89 |
| miru | 73 | 73 | 72 | 72 | 71 | 71 | 71 | 71 | 71 | 71 | 71 |
| mukaeru | 89 | 89 | 89 | 89 | 89 | 89 | 89 | 91 | 90 | 90 | 90 |
| motsu | 57 | 62 | 61 | 61 | 62 | 61 | 62 | 61 | 61 | 61 | 61 |
| motomeru | 87 | 87 | 87 | 87 | 87 | 87 | 87 | 87 | 87 | 87 | 87 |
| yomu | 88 | 88 | 88 | 88 | 88 | 88 | 88 | 88 | 88 | 88 | 88 |
| yoru | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 97 | 96 |
| wakaru | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 | 90 |
| 平均 | 78.16 | 78.44 | 78.82 | 79.00 | 79.16 | 79.16 | 79.16 | 79.06 | 78.92 | 78.82 | 78.70 |

## 4.3　素性選択の指標値

教師なし学習手法を用いる場合，どのような素性を使うかが重要である．そこで，本実験では，判別力の高さ (P) と Naive Bayes のモデルの仮定を満たす程度 (E) の 2 つの観点が素性選択に影響していると考え，それらを使って，素性選択の指標値を提案した．詳細は以下の通りである．

まず確率モデルどうしの距離を KL 情報量により測り，E を以下のように定義する．

$$E = \sum_{c} P(c) I(\prod_{i=1}^{n} P(f_i|c), P(x|c))$$

ここで I は真のモデル $p$ と比較対象のモデル $q$ との KL 情報量であり，以下で定義される．

$$I(p;q) = \sum_{i=1}^{N} p_i \log \frac{p_i}{q_i}$$

判別力の高さ (P) は，ラベル付き訓練データのみから学習できる Naive Bayes の分類器のラベル付き訓練データに対する精度に設定する．

E の値は小さいほどよく，P の値は大きいほどよい．P の値は正規化されているので，$E/P$ を素性選択の指標値とすることにした．

ただし現実的には素性の数が多いと，E の値は計算できない．以後の実験では，素性の数を 2 つに限定して，この指標値の妥当性を示した．

ここでは，単語「声」の語義判別規則の学習に EM アルゴリズムを適用する．「声」の語義は大きく『意見』と『喉から発声される音』という語義がある．本実験では曖昧性をなくすために『意見』という語義とそれ以外の語義という形で 2 つの語義を設定し，前者を a，後者を b とする．

次に毎日新聞の'95 年度 1 年分の記事を形態素解析し，「声」という単語を含む 7,041 文を取り出した．全部で 7,041 文存在した．次に，そこからランダムに 100 文と 300 文を取り出し，それらの各文の持つ「声」の語義に応じて a または b のラベルを付与した．ラベル付きの 100 文を最初のラベル付き訓練データ L とし，ラベル付きの 300 文を評価のためのテストデータ T とした．残りの 6,641 文がラベルなし訓練データ U である．EM アルゴリズムは 10 回の繰り返しで終了することにした．

素性 e1〜e4 を使った場合の結果を図 4.1 に示す．横軸は EM アルゴリズムの繰り返しの回数，縦軸は学習できた分類器の T に対する正解率 (%) である．最初のラベル付き訓練データだけから得た分類器の精度は 80.7%であったが，EM アルゴリズムを用いることで 84.0%まで向上した．

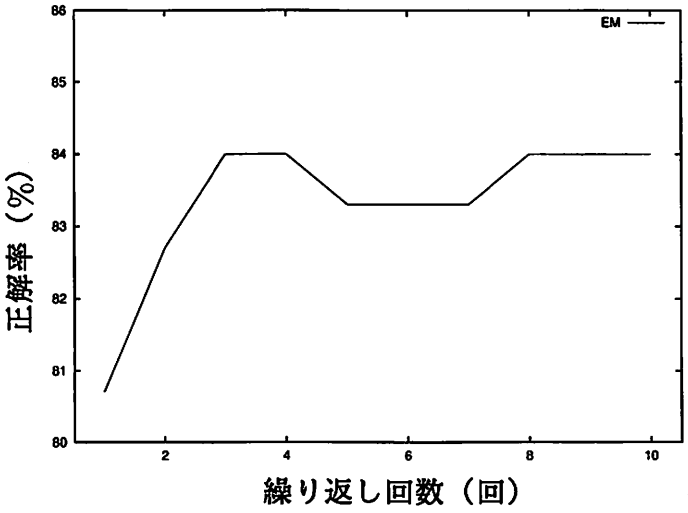


図 4.1: 「声」の語義判別の学習

次に素性 e1 ～ e4 のうちから 2つの素性だけを使った学習を行ってみる．まず利用する 2つの素性が Naive Bayes の仮定を満たす程度 (E) を測る．このために L から得たデータを使い，確率モデル推定した．現れない事例は頻度 0.1 にとった．また利用する 2つの素性の組の判別力 (P) を測った．これは L から Naive Bayes を用いた分類器を作成し，L における正解率である．以上より提案した素性選択の指標値 $E/P$ の値を算出した．次に各々の素性の組を用い，L のみから Naive Bayes による分類器の T に対する精度と，EM アルゴリズムを利用した教師なし学習を行って最終的に得られた分類器の T に対する精度を求めた．結果を表 4.5 にまとめる．表から $E/P$ の値が最も小さい e3 と e4 の組が，最も効果的に学習できていることがわかる．

表 4.5: 素性選択の指標値と正解率

| 素性 | E | P | E/P | Naive Bayes | Naive Bayes + EM |
|---|---|---|---|---|---|
| e1 + e2 | 0.412 | 0.85 | 0.485 | 79.3% | 77.3% |
| e1 + e3 | 0.671 | 0.98 | 0.685 | 82.3% | 85.3% |
| e1 + e4 | 0.593 | 0.92 | 0.645 | 84.3% | 85.3% |
| e2 + e3 | 0.628 | 0.96 | 0.654 | 76.7% | 77.0% |
| e2 + e4 | 0.673 | 0.91 | 0.740 | 78.0% | 75.7% |
| e3 + e4 | 0.460 | 0.99 | 0.464 | 78.7% | 86.3% |

## 4.4　k-最近傍法 + ファジィクラスタリング

名詞50単語に対して，単純な最近傍法（表中のNN），収束するまで繰り返し実行したファジィクラスタリング（表中のFuzzy1 ,... , Fuzzy5 ）の各結果を表4.6に示す．単純な最近傍法による分類で得られた正解率は76.83 % であった．それに対して，ファジィクラスタリングによる正解率は1回目が75.85 % となり，最近傍法の正解率を下回ってしまった．さらに，ファジィクラスタリングを繰り返すことで，正解率は徐々に向上してはいるが，最後まで最近傍法の正解率を上回ることはなかった．

同様に動詞50単語に対して，得られた結果が表4.7である．動詞による実験では，最近傍法の正解率が77.79 % であったが，ファジィクラスタリンでは78.01 % となり，最近傍法の正解率を上回った．ところが，ファジィクラスタリングを繰り返すことによって徐々に正解率が減少してしまった．

表 4.6: 実験結果（ファジィ：名詞）

| 単語 | NN | Fuzzy1 | Fuzzy2 | Fuzzy3 | Fuzzy4 | Fuzzy5 |
|---|---|---|---|---|---|---|
| aida | 79 | 71 | 71 | 71 | 71 | 71 |
| atama | 65 | 59 | 59 | 59 | 60 | 60 |
| ippan | 88.3332 | 89.6665 | 90.6665 | 90.6665 | 90.6665 | 90.6665 |
| ippou | 87 | 85 | 86 | 86 | 86 | 86 |
| ima | 85 | 72 | 72 | 72 | 72 | 72 |
| imi | 55 | 60 | 60 | 59 | 59 | 59 |
| utagai | 100 | 100 | 100 | 100 | 100 | 100 |
| otoko | 92 | 92 | 92 | 92 | 92 | 92 |
| kaihatsu | 64 | 64 | 64 | 64 | 64 | 64 |
| kaku_n | 72 | 73 | 73 | 73 | 73 | 73 |
| kankei | 84 | 86 | 86 | 86 | 86 | 86 |
| kimochi | 62 | 66 | 67 | 67 | 67 | 67 |
| kiroku | 63 | 58 | 60 | 63 | 63 | 63 |
| gijutsu | 96 | 95 | 95 | 95 | 95 | 95 |
| genzai | 98 | 98 | 98 | 98 | 98 | 98 |
| koushou | 100 | 100 | 100 | 100 | 100 | 100 |
| kokunai | 53 | 53 | 53 | 53 | 53 | 53 |
| kotoba | 51 | 51 | 50 | 50 | 50 | 50 |
| kodomo | 70 | 60 | 60 | 60 | 60 | 60 |
| gogo | 88.5 | 85.5 | 87.5 | 87.5 | 74 | 83.5 |
| shijo | 71 | 67 | 67 | 67 | 67 | 67 |
| shimin | 63 | 52 | 51 | 52 | 52 | 52 |
| shakai | 79 | 80 | 80 | 80 | 80 | 80 |
| shonen | 92 | 92 | 92 | 92 | 92 | 92 |
| jikan | 54 | 53 | 53 | 53 | 53 | 53 |
| jigyou | 68 | 72 | 72 | 72 | 72 | 72 |
| jidai | 72 | 77 | 77 | 77 | 77 | 77 |
| jibun | 100 | 96 | 96 | 96 | 96 | 96 |
| joho | 74 | 70 | 71 | 71 | 71 | 71 |
| sugata | 59 | 61 | 60 | 61 | 61 | 61 |
| seishin | 66 | 66 | 65 | 65 | 65 | 65 |
| taishou | 96 | 96 | 96 | 96 | 96 | 96 |
| daihyou | 85.5 | 89.5 | 89.5 | 89.5 | 89.5 | 89.5 |
| chikaku | 81 | 81 | 81 | 81 | 82 | 82 |
| chihou | 74 | 66 | 66 | 66 | 66 | 66 |
| chushin | 98 | 98 | 98 | 98 | 98 | 98 |
| te | 46 | 47 | 47 | 47 | 47 | 47 |
| teido | 99 | 99 | 99 | 99 | 99 | 99 |
| denwa | 85 | 85 | 85 | 85 | 85 | 85 |
| doujitsu | 65 | 66 | 66 | 68 | 70 | 71 |
| hana | 99 | 99 | 99 | 99 | 99 | 99 |
| hantai | 98 | 98 | 98 | 98 | 98 | 98 |
| baai | 90 | 86 | 86 | 86 | 86 | 86 |
| mae | 87 | 87 | 87 | 87 | 87 | 87 |
| minkan | 100 | 100 | 100 | 100 | 100 | 100 |
| musume | 86 | 84 | 84 | 84 | 84 | 84 |
| mune | 63 | 67 | 67 | 67 | 67 | 67 |
| me | 13 | 13 | 13 | 13 | 13 | 13 |
| mono | 30 | 32 | 32 | 32 | 32 | 32 |
| mondai | 95 | 95 | 95 | 95 | 95 | 95 |
| 平均 | 76.83 | 75.85 | 75.93 | 76.05 | 75.86 | 76.07 |

表 4.7: 実験結果（ファジィ：動詞）

| 単語 | NN | Fuzzy1 | Fuzzy2 | Fuzzy3 | Fuzzy4 | Fuzzy5 |
|---|---|---|---|---|---|---|
| ataeru | 68 | 68 | 69 | 66 | 68 | 68 |
| iu | 94 | 93 | 93 | 93 | 93 | 93 |
| ukeru | 49 | 59 | 62 | 62 | 62 | 62 |
| uttaeru | 83 | 86 | 87 | 88 | 88 | 88 |
| umareru | 68 | 70 | 70 | 70 | 70 | 70 |
| egaku | 57 | 53 | 53 | 53 | 53 | 53 |
| omou | 90 | 91 | 91 | 91 | 91 | 91 |
| kau | 85 | 86 | 86 | 85 | 85 | 85 |
| kakaru | 60 | 61 | 62 | 64 | 64 | 64 |
| kaku_v | 75 | 74 | 74 | 74 | 74 | 74 |
| kawaru | 92 | 92 | 92 | 92 | 92 | 92 |
| kangaeru | 99 | 99 | 99 | 99 | 99 | 99 |
| kiku | 65 | 64 | 63 | 63 | 63 | 63 |
| kimaru | 96 | 96 | 96 | 96 | 96 | 96 |
| kimeru | 92 | 93 | 93 | 92 | 91 | 91 |
| kuru | 85 | 82 | 82 | 82 | 82 | 82 |
| kuwaeru | 89 | 89 | 89 | 89 | 89 | 89 |
| koeru | 78 | 84 | 84 | 84 | 84 | 84 |
| shiru | 97 | 97 | 97 | 97 | 97 | 97 |
| susumu | 42 | 53 | 51 | 52 | 52 | 52 |
| susumeru | 97 | 96 | 96 | 96 | 96 | 96 |
| dasu | 30 | 31 | 31 | 31 | 31 | 31 |
| chigau | 100 | 100 | 100 | 100 | 100 | 100 |
| tsukau | 95.5 | 85.75 | 85.75 | 85.75 | 85.75 | 85.75 |
| tsukuru | 66 | 64 | 64 | 64 | 64 | 64 |
| tsutaeru | 75 | 76 | 76 | 76 | 76 | 76 |
| dekiru | 81 | 80 | 80 | 78 | 78 | 78 |
| deru | 54 | 52 | 52 | 52 | 52 | 52 |
| tou | 71 | 69 | 70 | 68 | 68 | 68 |
| toru | 32 | 35 | 35 | 34 | 33 | 33 |
| nerau | 99 | 99 | 99 | 99 | 99 | 99 |
| nokosu | 79 | 76 | 75 | 75 | 75 | 75 |
| noru | 57 | 61 | 61 | 61 | 61 | 61 |
| hairu | 37 | 39 | 39 | 39 | 39 | 39 |
| hakaru | 93 | 93 | 93 | 93 | 93 | 93 |
| hanasu | 100 | 100 | 100 | 100 | 100 | 100 |
| hiraku | 85 | 88 | 87 | 87 | 87 | 87 |
| fukumu | 99 | 97 | 97 | 97 | 97 | 97 |
| matsu | 44 | 52 | 52 | 52 | 52 | 52 |
| matomeru | 80 | 75 | 73 | 73 | 72 | 72 |
| mamoru | 79 | 75.5 | 73.5 | 73.5 | 73.5 | 73.5 |
| miseru | 97 | 97 | 97 | 96 | 96 | 96 |
| mitomeru | 89 | 89 | 89 | 89 | 89 | 89 |
| miru | 81 | 78 | 78 | 78 | 78 | 78 |
| mukaeru | 89 | 89 | 89 | 89 | 89 | 89 |
| motsu | 55 | 51 | 51 | 52 | 52 | 52 |
| motomeru | 87 | 87 | 87 | 87 | 87 | 87 |
| yomu | 88 | 88 | 88 | 88 | 88 | 88 |
| yoru | 96 | 97 | 97 | 97 | 97 | 97 |
| wakaru | 90 | 90 | 90 | 90 | 90 | 90 |
| 平均 | 77.79 | 78.01 | 77.97 | 77.85 | 77.83 | 77.83 |

# 第5章　考察

本実験により，語義判別問題に対しても，クラスタリング使った教師なし学習が適用可能であることが示された．ある種の単語に対しては，この手法により精度向上が図れる．

EM アルゴリズムを使った教師なし学習では約半数の精度が向上された．しかし精度が逆に悪くなるケースも存在した．実験 e1 と e2 だけで学習させると，精度は悪くなる．ところが，この 2つの素性の組は Naive Bayes の仮定を満たす程度 (E) が最も良い．つまり単純に E の値だけでは利用すべき素性を判断できない．また単純に想定した素性をすべて使えば正解率が向上するわけでもない．そのために頑健性の高い素性の選択法は重要な課題である．また，本実験では 2つの素性に限定して，Naive Bayes の仮定を満たす程度 (E) と，素性の組の判断力 (P) から素性選択の指標値 ($E/P$ ) を提案したが，

本論文で用いた教師なし学習の手法は，利用する素性が 3つ以上でも全く構わない．その場合，どのようにして E を測ればよいかは今後の課題である．また E と P が素性選択に関わっていると思われるが，指標値を $E/P$ の形で与えることが適切かどうかも今後検討する必要がある．例えば，素性 e1，e3，e4 の 3つを使った教師なし学習の結果を表 5.1 に示す．これが今回行った実験の中でも最もよい値を出した．Naive Bayes の仮定を満たす程度から考えれば，素性 e1，e3，e4 の 3つ組はよい選択とは思えないので，$E/P$ では適切な指標値を与えられていない．

表 5.1: 精度の高い素性の組

| 素性 | Naive Bayes | Naive Bayes + EM |
|---|---|---|
| e1 + e2 + e3 + e4 | 80.7% | 84.0% |
| e3 + e4 | 78.7% | 86.3% |
| e1 + e3 + e4 | 83.3% | 87.0% |

EM アルゴリズムを用いた手法は複数観点を利用している．複数観点を利用した手法としては，他に Co-training [1] がある．この 2つの手法を比べた場合，Co-training は独立な 2つの素性集合を設定しなければならないという問題がある．一方，EM アルゴリズムを利用した手法は，データの発生源や素性に課しているモデルが厳しい．このため Co-training

の方が応用範囲が広く現実的な手法と言える．また文書分類に限れば，Co-training の方が EM アルゴリズムを利用した手法よりも優れていたという報告もある [2]．しかし語義判別のような多値分類問題を扱う場合，Co-training は独立な 2 つの素性集合という問題以外に，素性の一貫性という条件が必要であり，その適用は難しい．一方，EM アルゴリズムを利用した手法は，原理的には分類問題が多値であっても影響はない．このため多値の分類問題への適用の観点から見れば，EM アルゴリズムを利用した手法の方が現実的である．

ファジィクラスタリングも全体の精度はほぼ変化がなかったと言える．これは本手法の効果が低いことを意味してはいない．それは個々の単語で見ると，正解率が大きく向上するものや大きく低下する単語が存在するからである．例えば，ukeru（図 5.1），susumu（図 5.2），matsu（図 5.3），doujitsu（図 5.4）などは大きく正解率が向上しているが，ima（図 5.5），shimin（図 5.6），kodomo（図 5.7），tsukau（図 5.8）などは大きく正解率が低下している．つまり本手法はある単語については効果があるが，ある単語については逆効果になっている．

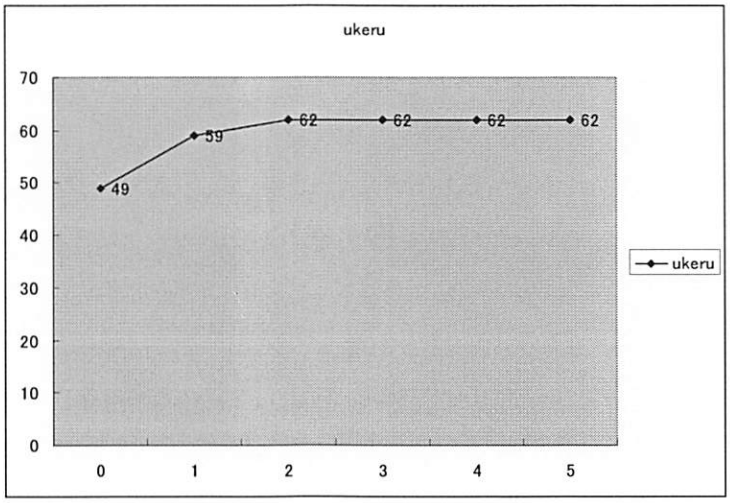


図 5.1: 正解率が向上した単語（ukeru）

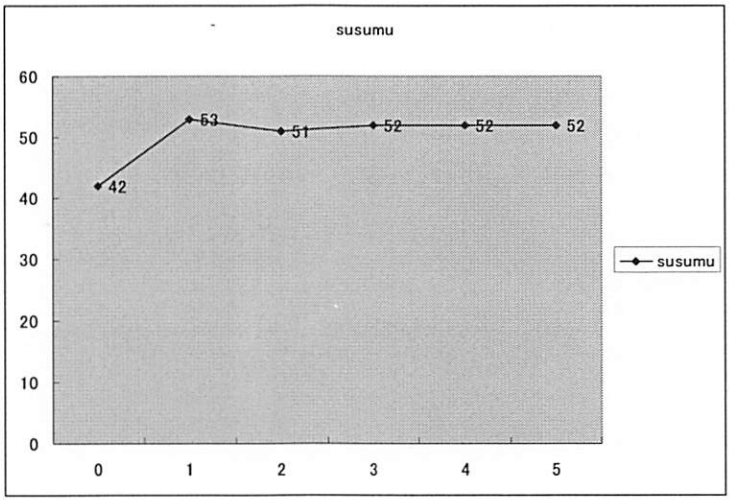


図 5.2: 正解率が向上した単語（susumu）

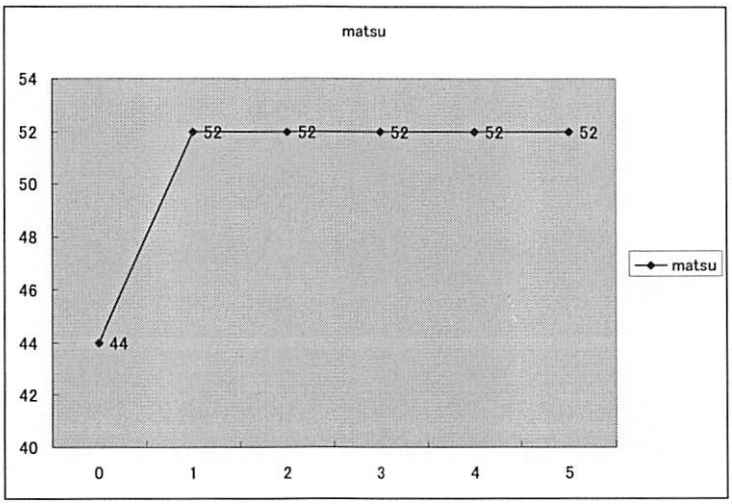


図 5.3: 正解率が向上した単語（matsu）

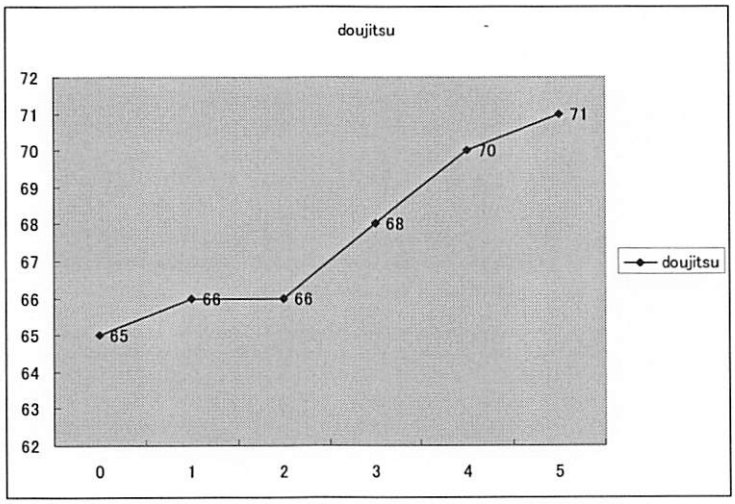


図 5.4: 正解率が向上した単語（doujitsu）

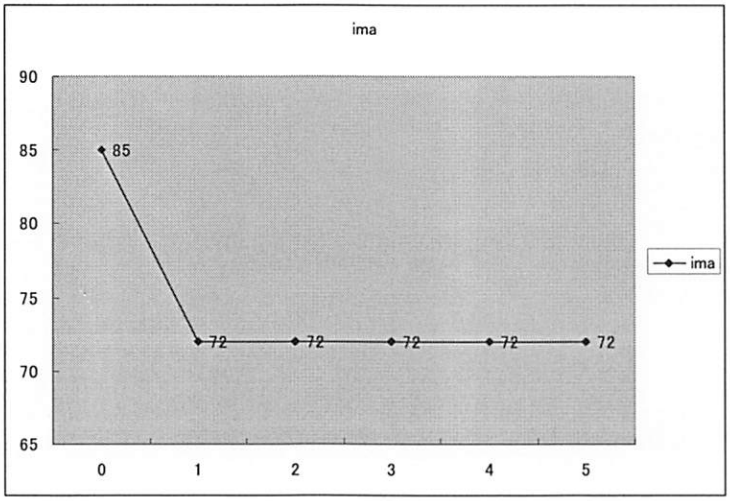


図 5.5: 正解率が低下した単語（ima）

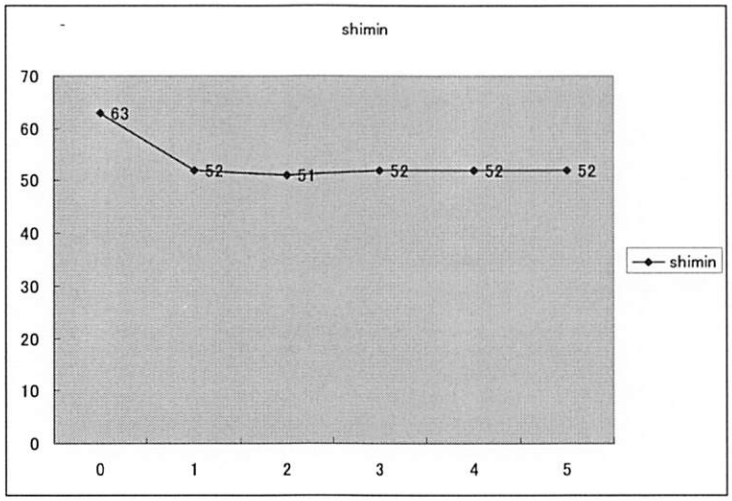


図 5.6: 正解率が低下した単語（shimin）

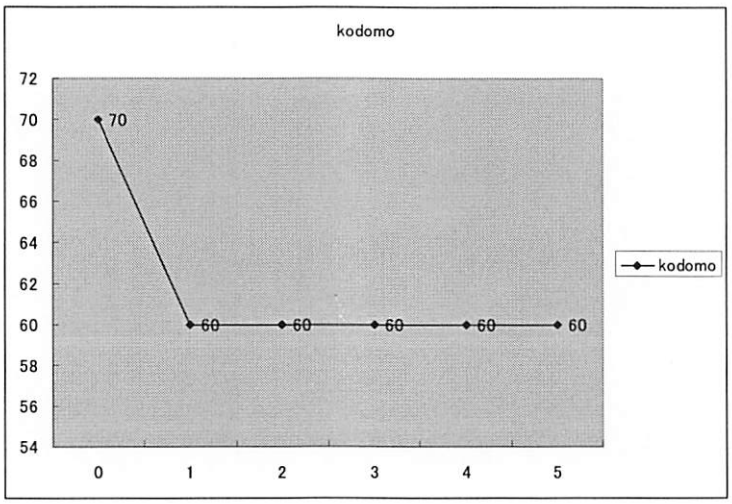


図 5.7: 正解率が低下した単語（kodomo）

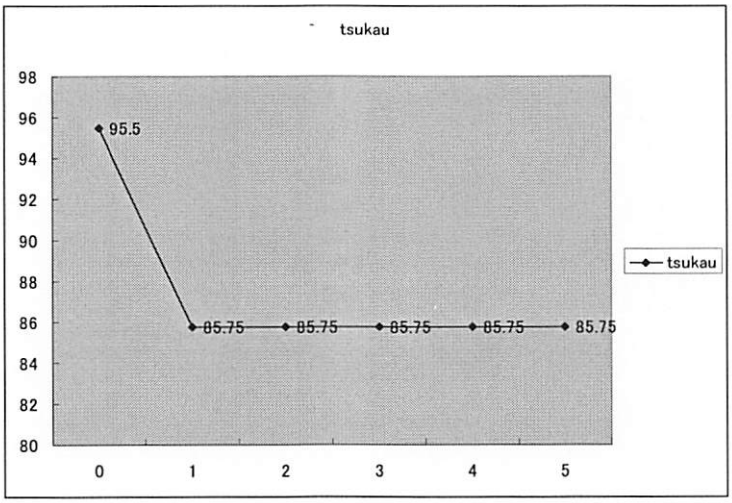


図 5.8: 正解率が低下した単語（tsukau）

ラベル付き訓練データの他にラベルなし訓練データを用いる教師なし学習は，ラベル付き訓練データのみを用いる通常の学習よりも，精度が低くなる場合もある．これは同じデータで教師なし学習を行った研究でも確認されている [6]．この場合の対処法としては2つあると考える．1つは論文 [6] のように効果がある問題とない問題を予め予測して，適用の有無や程度を調整するアプローチである．もう1つは副作用が少ない，つまり頑健性が高い教師なし学習を用いることである．そしてこのファジィクラスタリングは比較的頑健性が高いと思われる．

EMアルゴリズムを用いた手法と比較した場合，何の対処も行わずにそのまま適用すると，名詞は 77.54% から 72.82% に正解率が下がり，動詞は 78.44 % から 78.70 % に正解率が上がる．全体としては 77.99% から 75.76%，つまり 2.23% の正解率の低下があるが，本手法の正解率の低下は 0.34% である．これは本手法で極端に正解率が悪くなることがないことを示している．

また本手法により得られたプロトタイプは新たな事例として扱えるという長所も持つ．このため，様々な展開が考えられる．例えば，得られた事例から別の学習手法を用いて分類器を作成することも可能であろう．

本手法の欠点としては計算時間が膨大になる点がある．アルゴリズム FCM はプロトタイプの数（つまりラベル付き訓練データの数）が多いと計算時間が多大にかかる．本実験では Pentium-4 1.5GHz のマシンで，1回の繰り返しに 1〜2 時間はかかった．計算を効率よく行う工夫は今後の課題である．

またベースとした k-最近傍法自体あまり識別精度が高くなかったので，この点も改良しなくてはならない．とくに事例をベクトルで表現する際に，素性のあるなしで 1 か 0 の値をつけるのは適切ではない．素性の重みを考慮すべきである．また k の値をここでは 5 にしたが，単語によって最適な k の値は異なるであろう．これらの点からの改良も今後の課題である．

# 第6章　おわりに

本論文ではEMアルゴリズム，およびファジィクラスタリングを語義の多義性の曖昧性解消問題に適用した．SENSEVAL2の日本語辞書タスクを用いた実験では，両手法とも全体的な精度の向上は見受けられなかった．個別に見ると精度が向上している単語も見受けられたものの，多くはある地点まで精度が上がっても，最終的にはそれよりも低い精度で収束してしまうか，悪くすると，ラベル付きの訓練事例のみから学習された規則よりも低い精度で収束してしまった．今後は最適な繰り返し回数を予測する何らかの方法を導入し，繰り返しによる精度の低下を防ぐことで，より精度の高いクラスタリングを実現したい．

# 謝辞

本研究の遂行および論文作成において多大な御助言及び御指導を賜りました新納 浩幸 教官（茨城大学工学部システム工学科）に深い感謝の意を表します．

また，その他様々な助言を頂きました主査の石黒 美佐子 教官，御指導を頂きましたシステム工学科計算機応用学講座の教官の方々，同研究室の阿部 修也 氏（茨城大学大学院理工学研究科システム工学専攻），紺野 憲一 氏（2002年度），山村 一起 氏（2002年度），藤枝 雅一 氏（2002年度），杉田 尚士 氏（2001年度），田邊 繁 氏（2001年度），進藤 修 氏（2001年度）に深く感謝致します．

# 参考文献


[1] Avrim Blum and Tom Mitchell. Combining Labeled andUnlabeled Data with Co-Training. In *11th Annual Conference on Computational Learning Theory(COLT-98)*, pp.92-100, 1998.

[2] Kamal Nigam and Rayid Ghani. Analyzing the effectiveness and aoolicability of co-training. In *9th International Conference on Information and Knowledge Management*, pp.86-93, 2000.

[3] Kamal Nigam, Andrew McCallum,Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. In *Machine Lerning*, Vol.39, pp.103-134, 2000.

[4] Tom Mitchell. *Machine Learning*. McGraw-Hill Companies, 1997.

[5] 黒橋禎夫，白井清昭．SENSEVAL-2 日本語タスク．電子情報通信学会，言語とコミュニケーション，NCL2001-36~48，pp.1-8，2001.

[6] 新納浩幸，佐々木稔．EM アルゴリズムの最適ループ回数の予測を用いた語義判別規則の教師なし学習．情報処理学会自然言語処理学会，NL-151-8，2002.

[7] 新納浩幸，髙橋篤史．EM アルゴリズムを用いた語義判別規則の教師なし学習．言語処理学会第 8 回年次大会，pp.663-666，2002.

[8] 新納浩幸，髙橋篤史．ファジィクラスタリングを用いた語義判別規則の教師なし学習．言語処理学会第 9 回年次大会，to appear，2003.

[9] 宮本定明．クラスター分析入門．北森出版，pp.1-64，1999.

[10] 国立国語研究所．分類語彙表．秀英出版，1994.

[11] 北研二．確率言語モデル．東京大学出版会，pp.41-46，1999.

# 付 録A　各単語の正解率

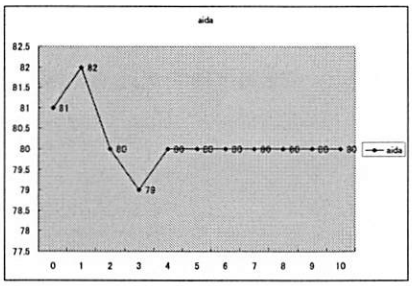


図 A.1: EM aida

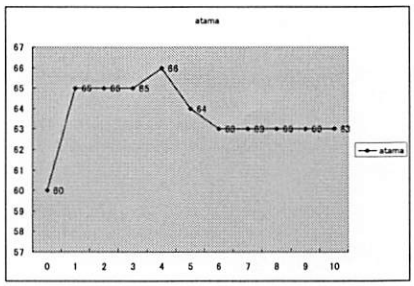


図 A.2: EM atama

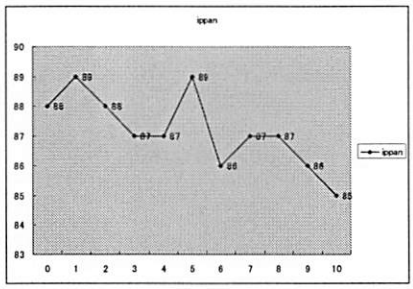


図 A.3: EM ippan

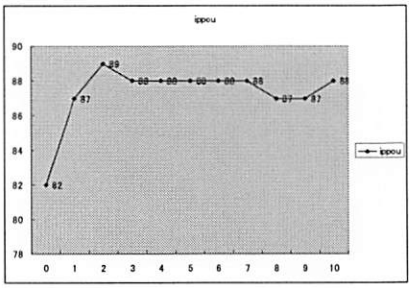


図 A.4: EM ippou

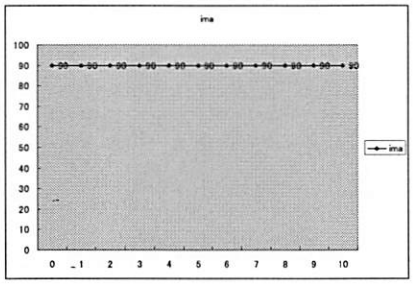


図 A.5: EM ima

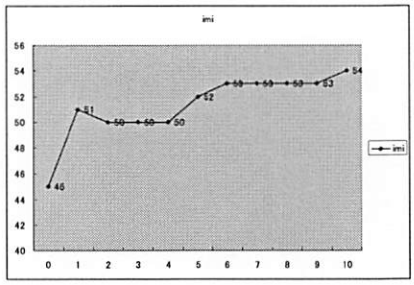


図 A.6: EM imi

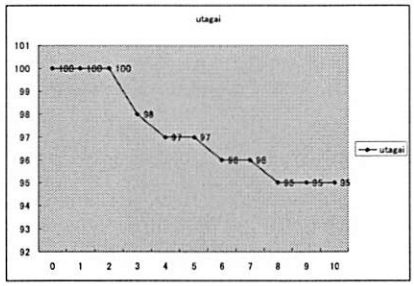


図 A.7: EM utagai

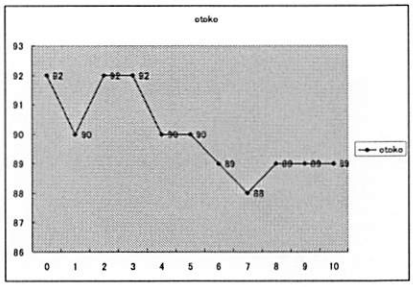


図 A.8: EM otoko

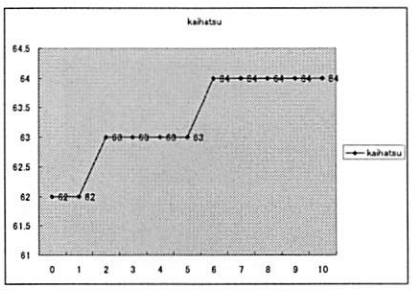


図 A.9: EM kaihatsu

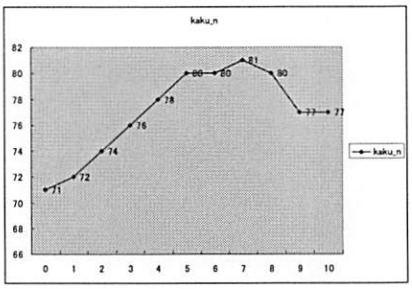


図 A.10: EM kaku_n

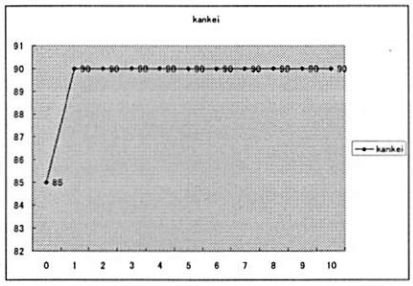


図 A.11: EM kankei

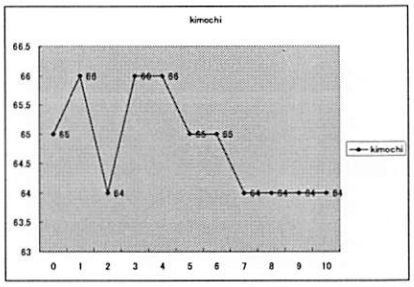


図 A.12: EM kimochi

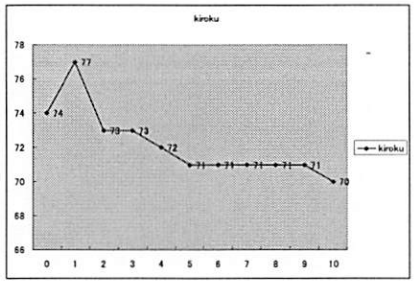


図 A.13: EM kiroku

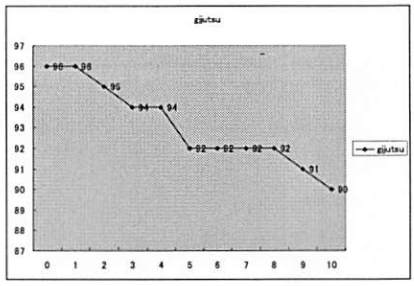


図 A.14: EM gijutsu

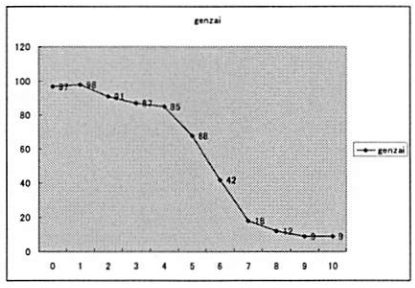


図 A.15: EM genzai

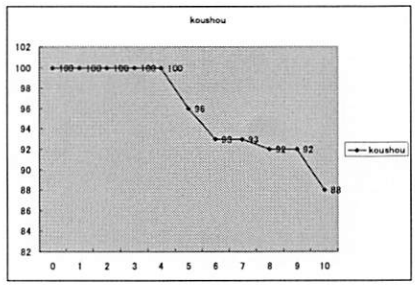


図 A.16: EM koushou

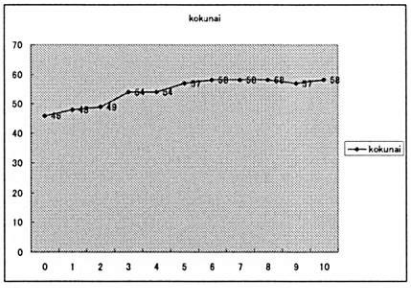


図 A.17: EM kokunai

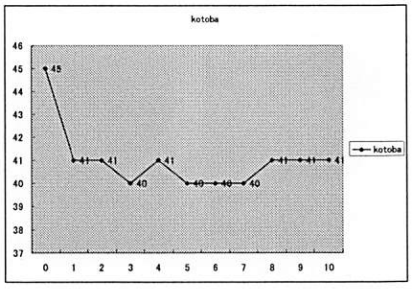


図 A.18: EM kotoba

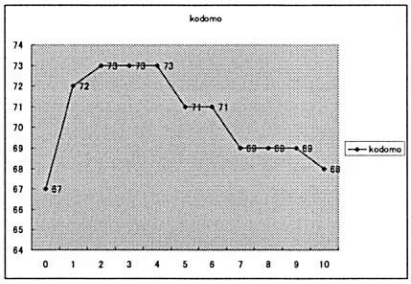


図 A.19: EM kodomo

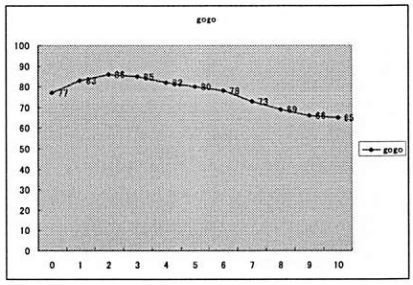


図 A.20: EM gogo

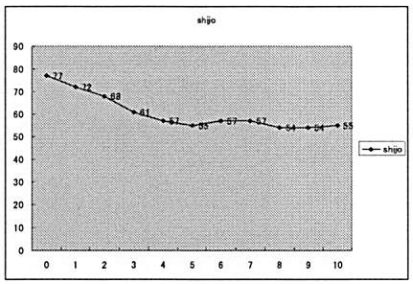


図 A.21: EM shijo

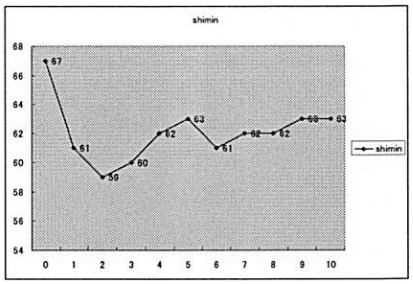


図 A.22: EM shimin

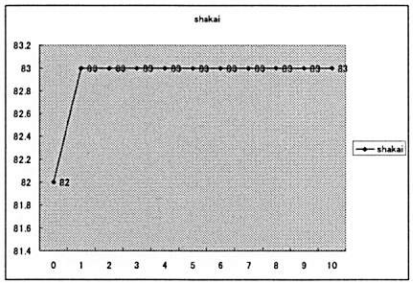


図 A.23: EM shakai

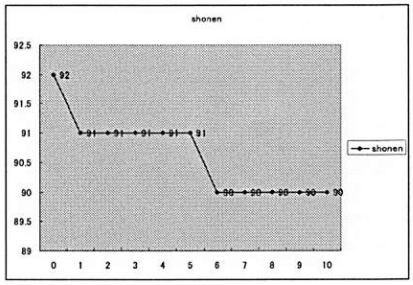


図 A.24: EM shonen

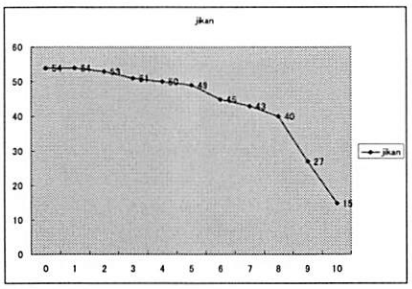


図 A.25: EM jikan

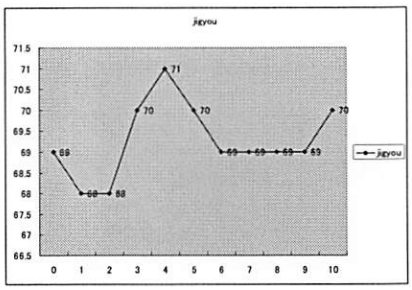


図 A.26: EM jigyou

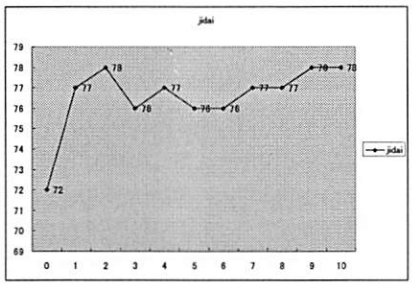


図 A.27: EM jidai

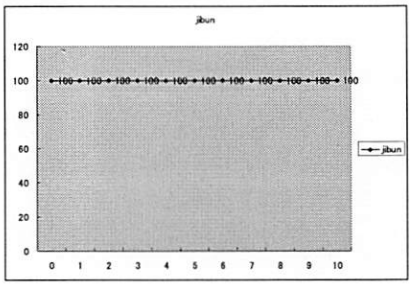


図 A.28: EM jibun

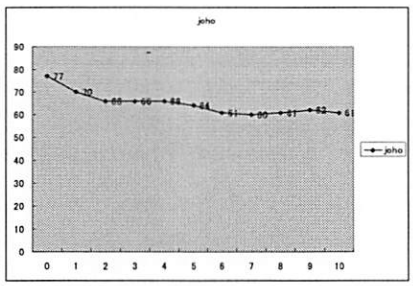


図 A.29: EM joho

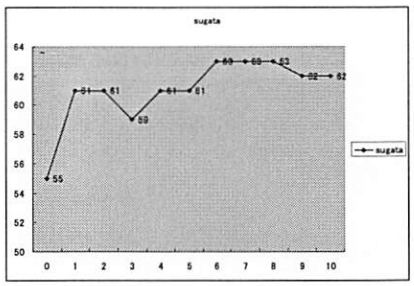


図 A.30: EM sugata

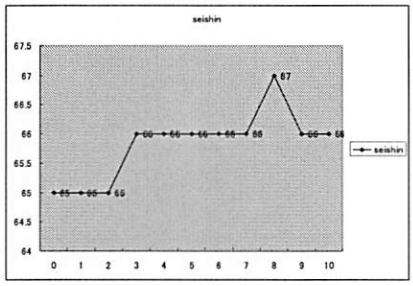


図 A.31: EM seishin

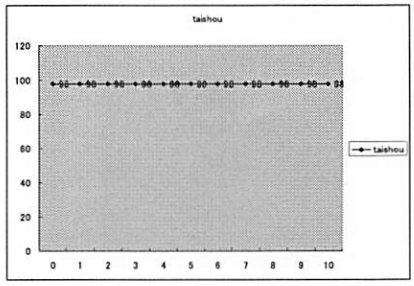


図 A.32: EM taishou

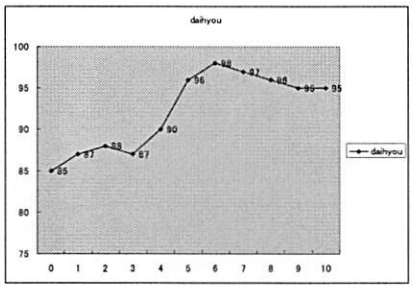


図 A.33: EM daihyou

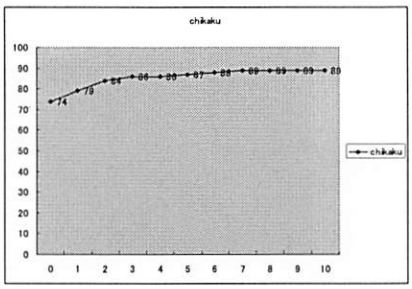


図 A.34: EM chikaku

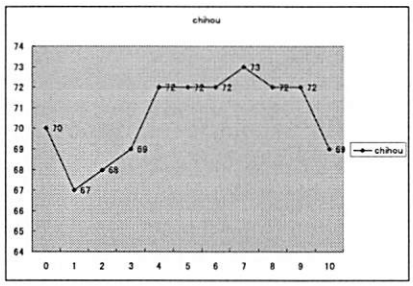


図 A.35: EM chihou

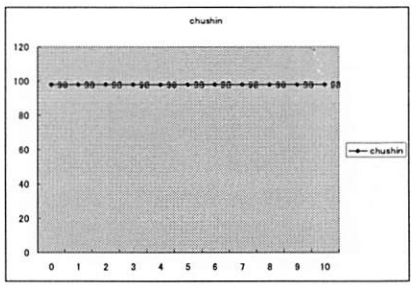


図 A.36: EM chushin

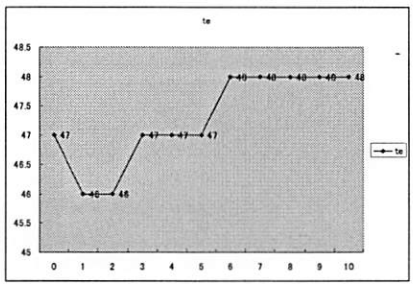


図 A.37: EM te

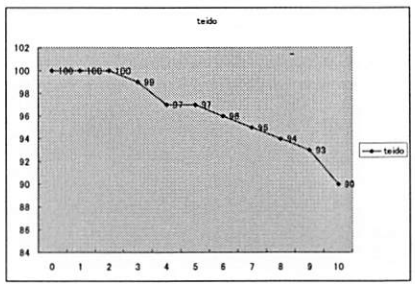


図 A.38: EM teido

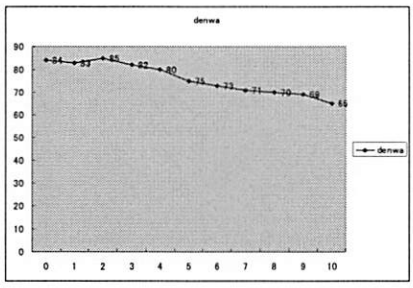


図 A.39: EM denwa

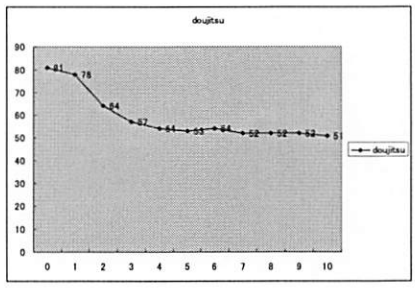


図 A.40: EM doujitsu

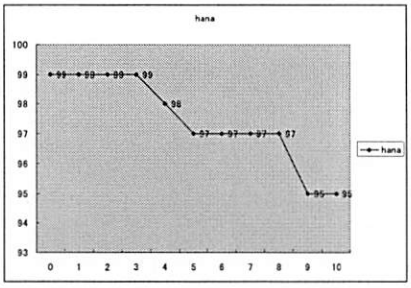


図 A.41: EM hana

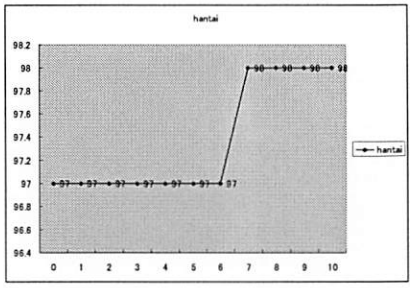


図 A.42: EM hantai

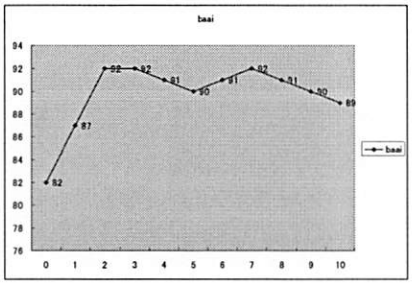


図 A.43: EM baai

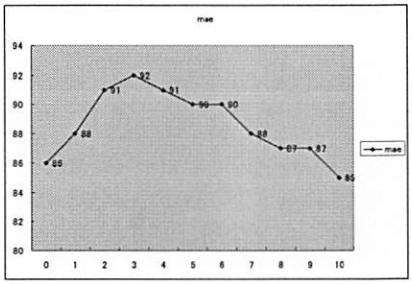


図 A.44: EM mae

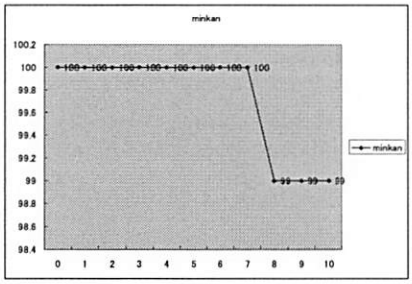


図 A.45: EM minkan

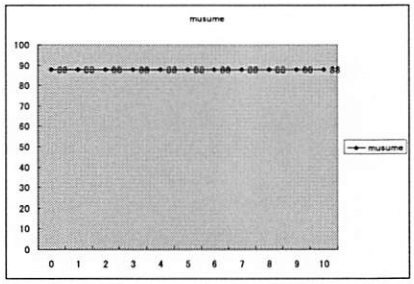


図 A.46: EM musume

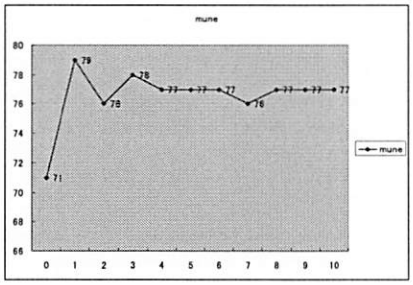


図 A.47: EM mune

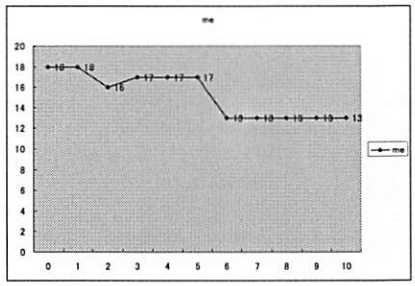


図 A.48: EM me

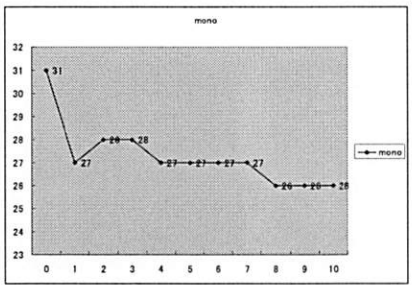


図 A.49: EM mono

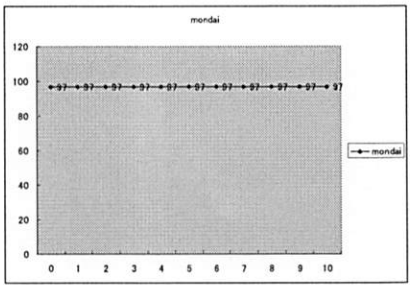


図 A.50: EM mondai

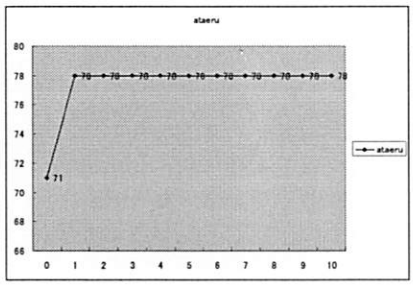


図 A.51: EM ataeru

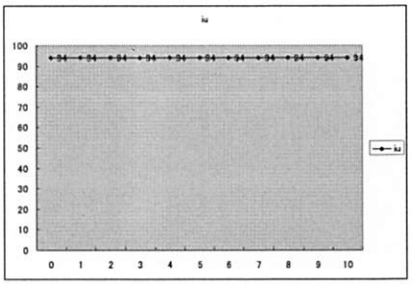


図 A.52: EM iu

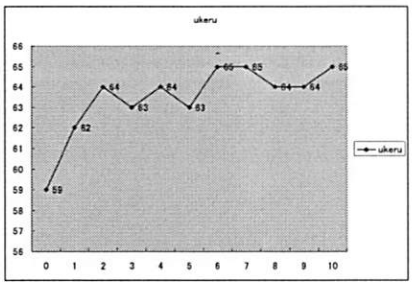


図 A.53: EM ukeru

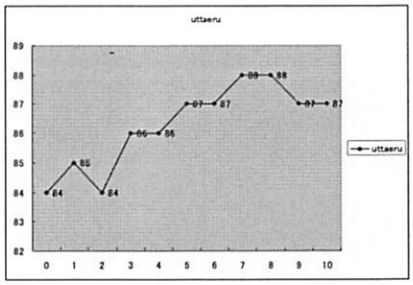


図 A.54: EM uttaeru

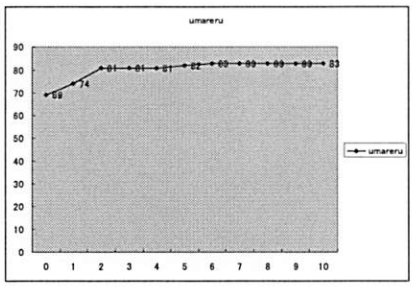


図 A.55: EM umareru

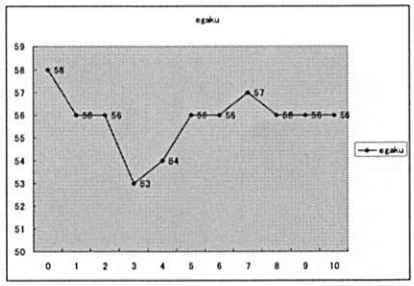


図 A.56: EM egaku

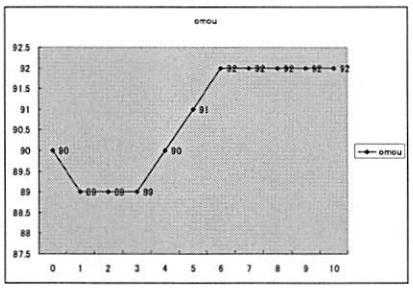


図 A.57: EM omou

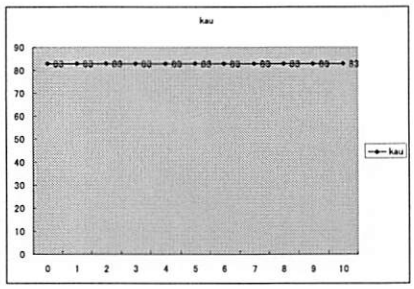


図 A.58: EM kau

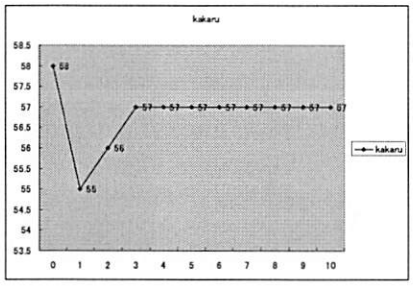


図 A.59: EM kakaru

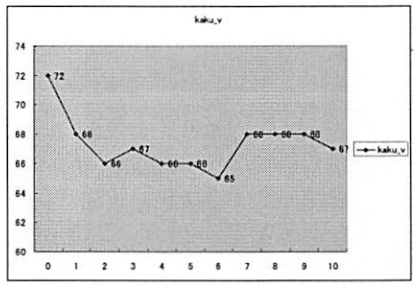


図 A.60: EM kaku_v

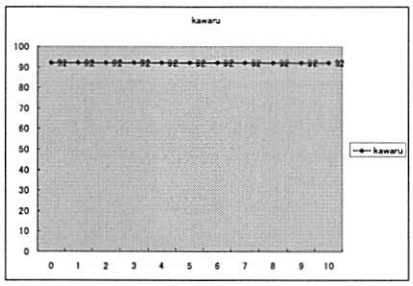


図 A.61: EM kawaru

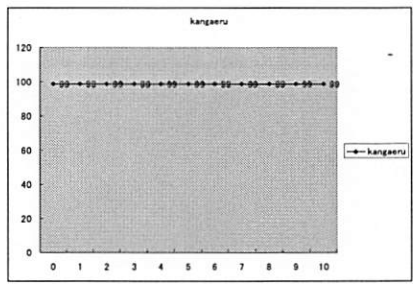


図 A.62: EM kangaeru

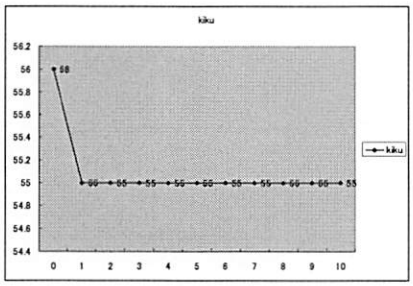


図 A.63: EM kiku

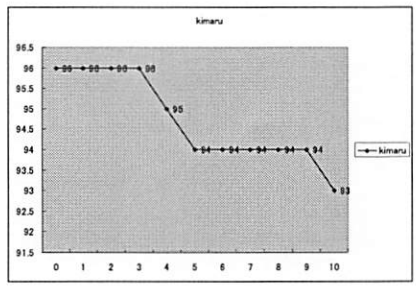


図 A.64: EM kimaru

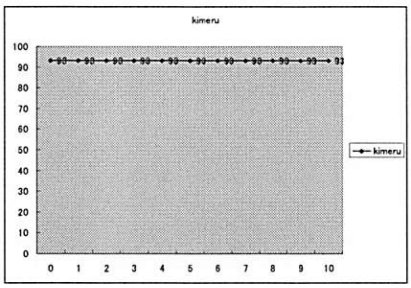


図 A.65: EM kimeru

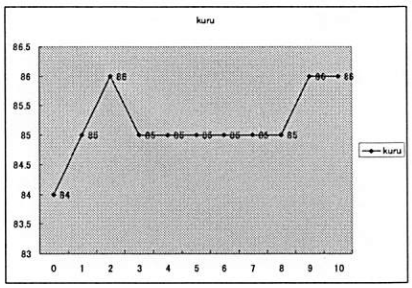


図 A.66: EM kuru

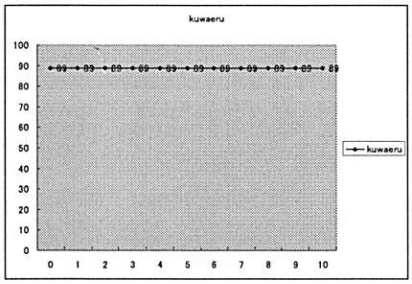


図 A.67: EM kuwaeru

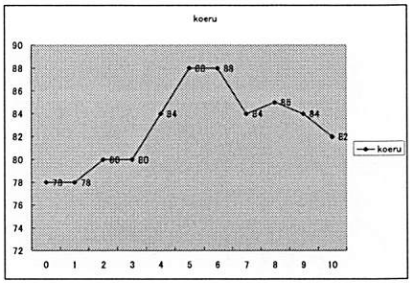


図 A.68: EM koeru

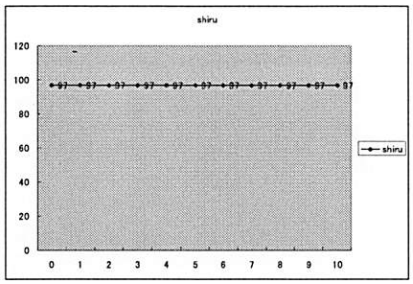


図 A.69: EM shiru

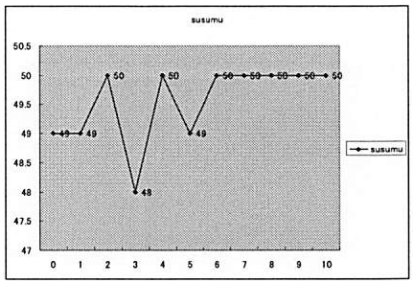


図 A.70: EM susumu

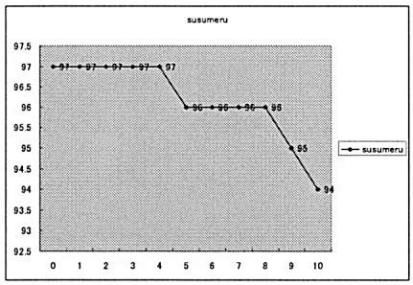


図 A.71: EM susumeru

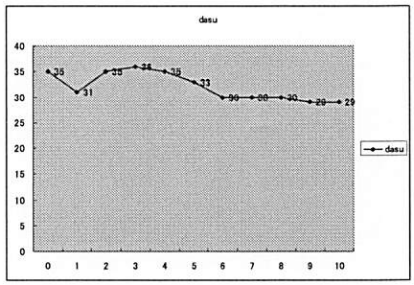


図 A.72: EM dasu

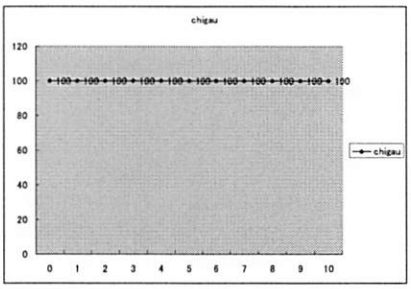


図 A.73: EM chigau

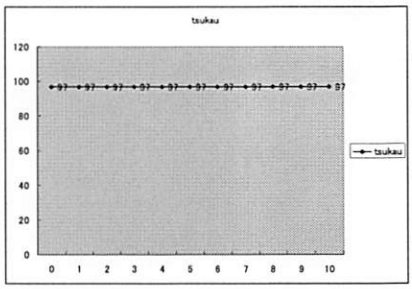


図 A.74: EM tsukau

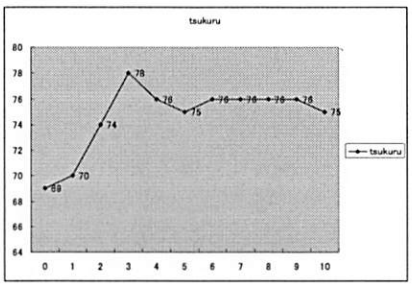


図 A.75: EM tsukuru

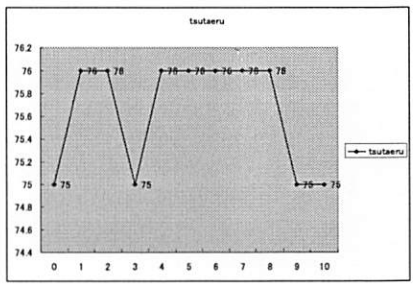


図 A.76: EM tsutaeru

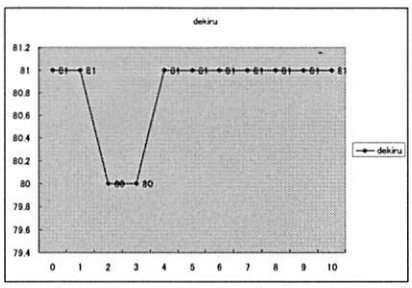


図 A.77: EM dekiru

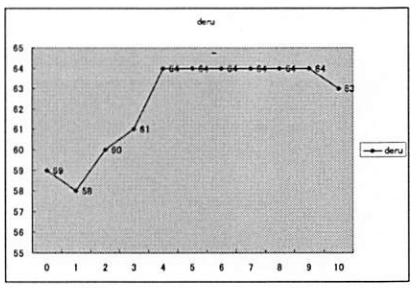


図 A.78: EM deru

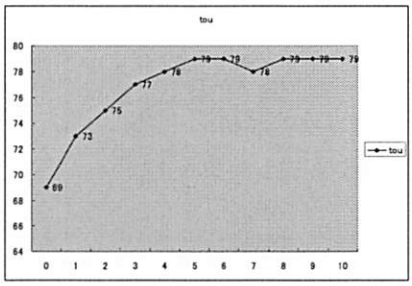


図 A.79: EM tou

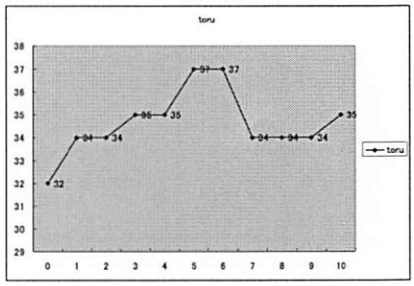


図 A.80: EM toru

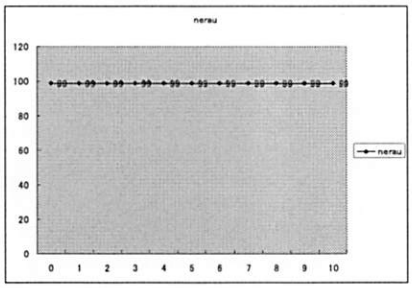


図 A.81: EM nerau

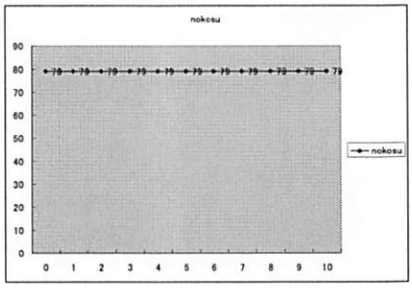


図 A.82: EM nokosu

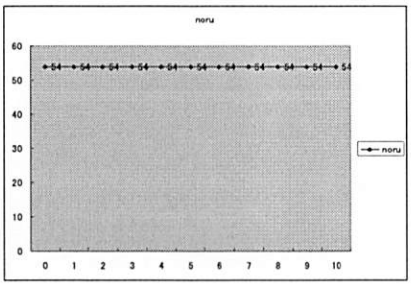


図 A.83: EM noru

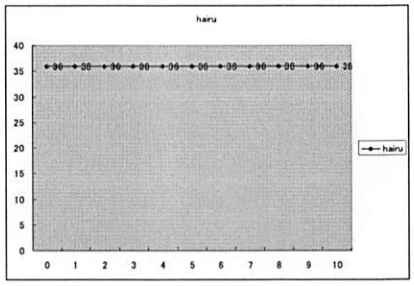


図 A.84: EM hairu

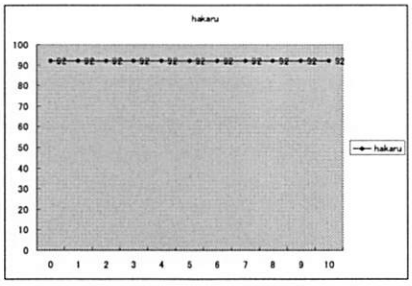


図 A.85: EM hakaru

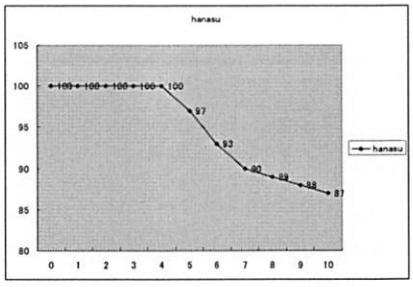


図 A.86: EM hanasu

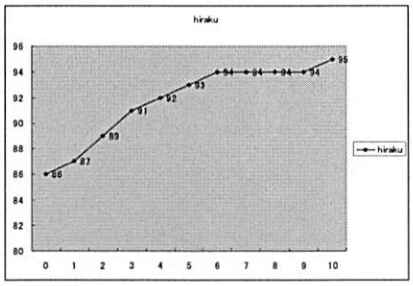


図 A.87: EM hiraku

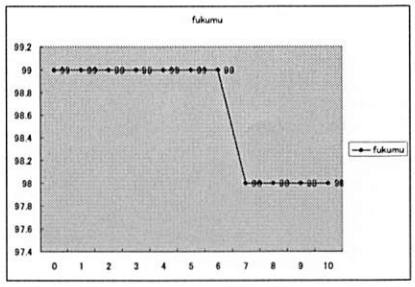


図 A.88: EM fukumu

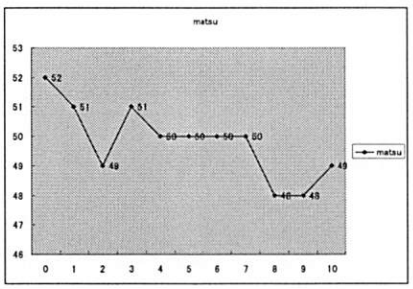


図 A.89: EM matsu

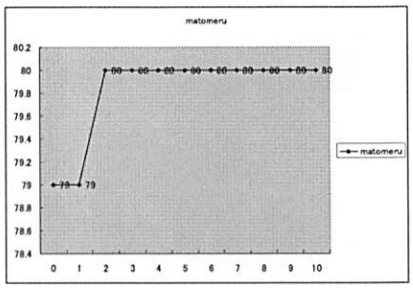


図 A.90: EM matomeru

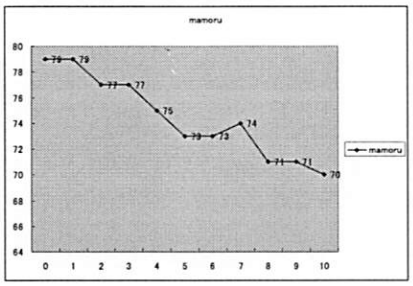


図 A.91: EM mamoru

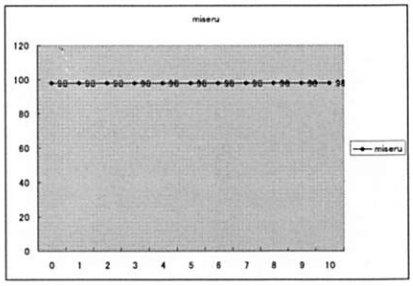


図 A.92: EM miseru

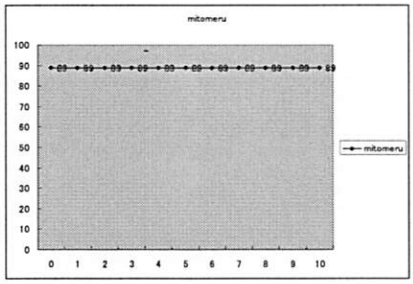


図 A.93: EM mitomeru

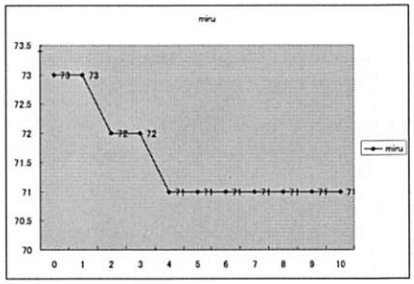


図 A.94: EM miru

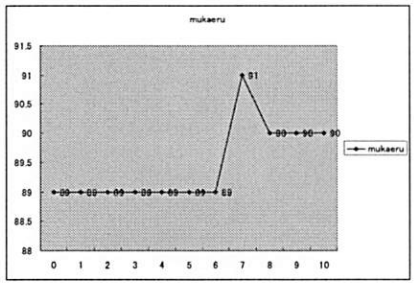


図 A.95: EM mukaeru

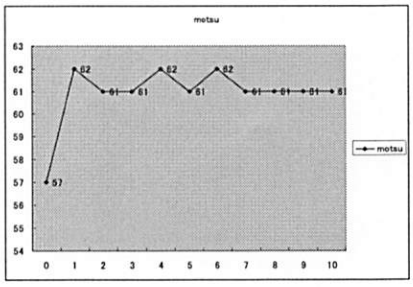


図 A.96: EM motsu

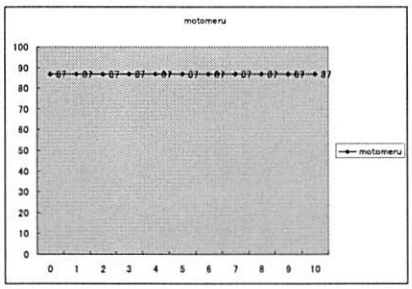


図 A.97: EM motomeru

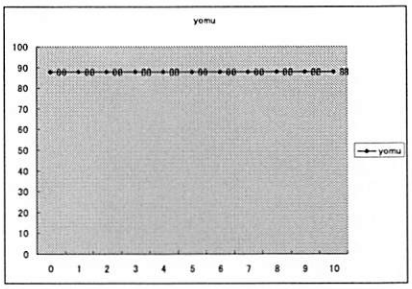


図 A.98: EM yomu

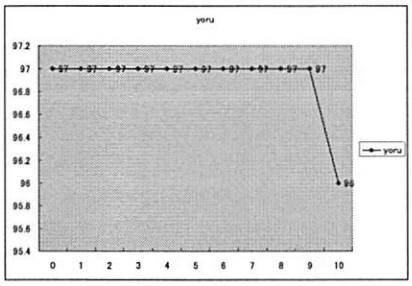


図 A.99: EM yoru

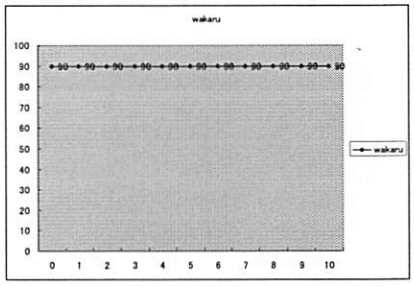


図 A.100: EM wakaru

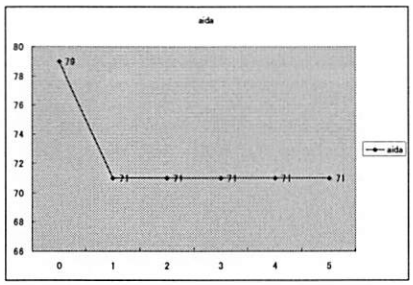


図 A.101: Fuzzy aida

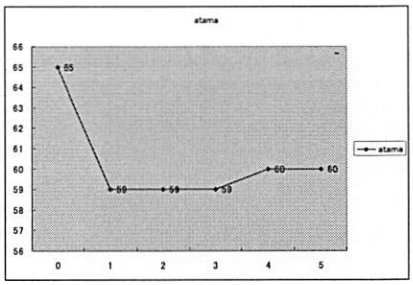


図 A.102: Fuzzy atama

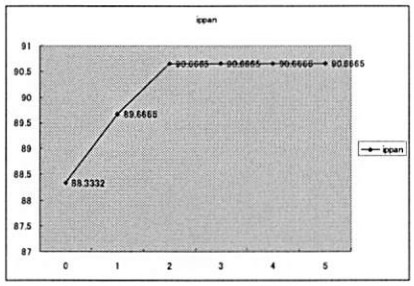


図 A.103: Fuzzy ippan

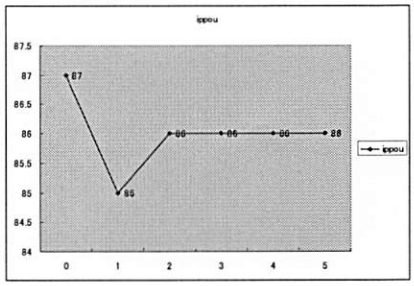


図 A.104: Fuzzy ippou

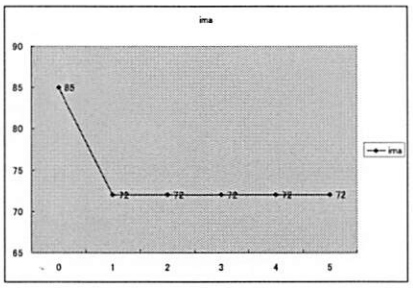


図 A.105: Fuzzy ima

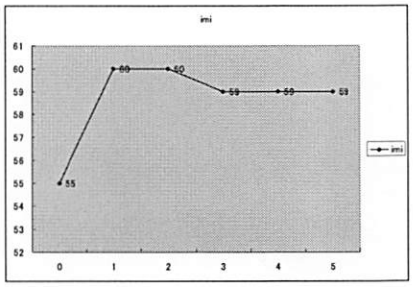


図 A.106: Fuzzy imi

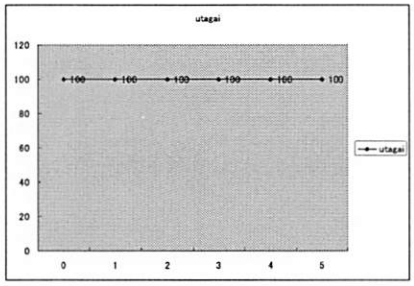


図 A.107: Fuzzy utagai

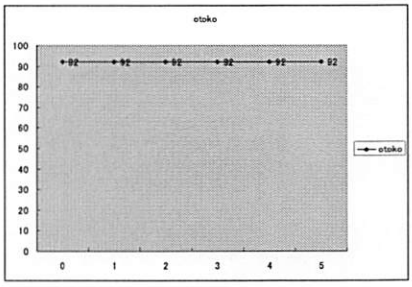


図 A.108: Fuzzy otoko

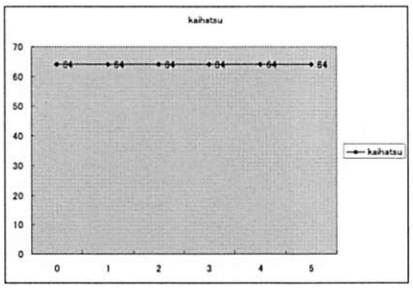


図 A.109: Fuzzy kaihatsu

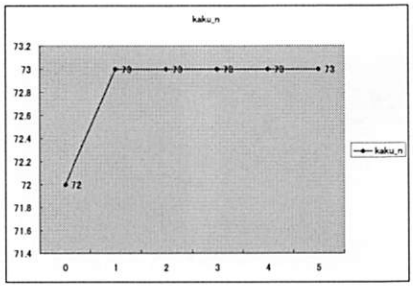


図 A.110: Fuzzy kaku_n

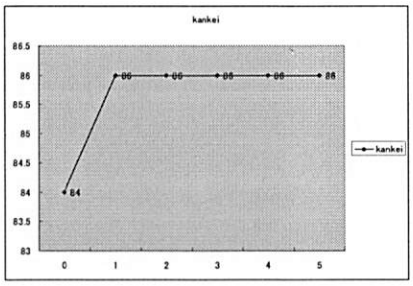


図 A.111: Fuzzy kankei

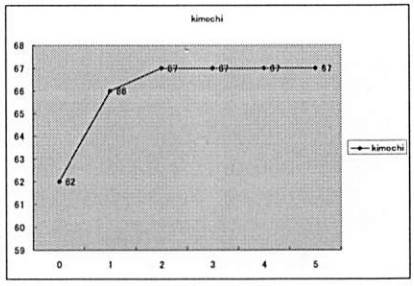


図 A.112: Fuzzy kimochi

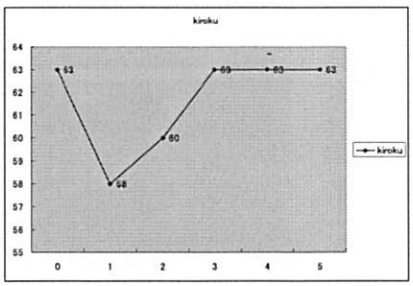


図 A.113: Fuzzy kiroku

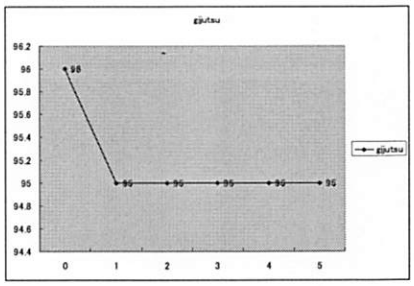


図 A.114: Fuzzy gijutsu

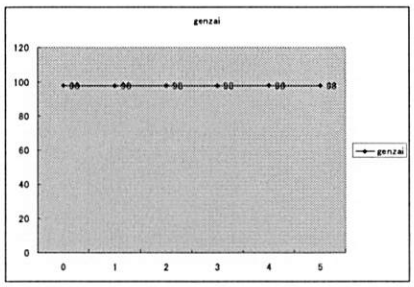


図 A.115: Fuzzy genzai

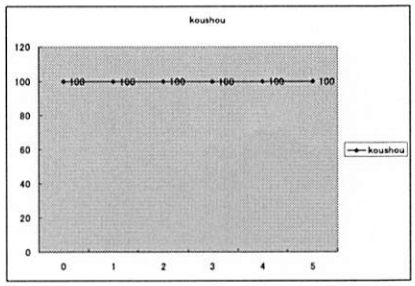


図 A.116: Fuzzy koushou

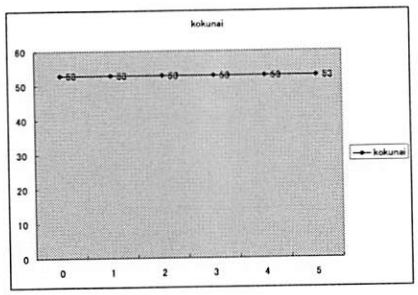


図 A.117: Fuzzy kokunai

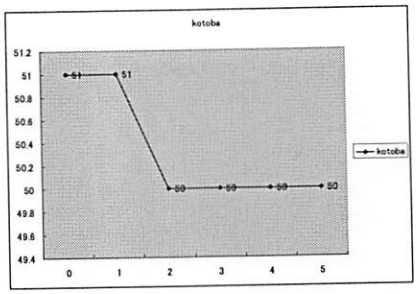


図 A.118: Fuzzy kotoba

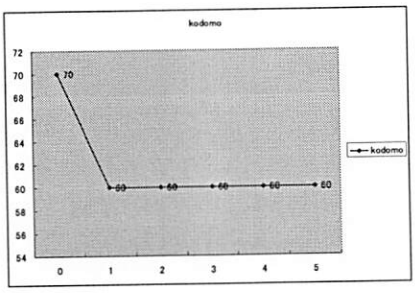


図 A.119: Fuzzy kodomo

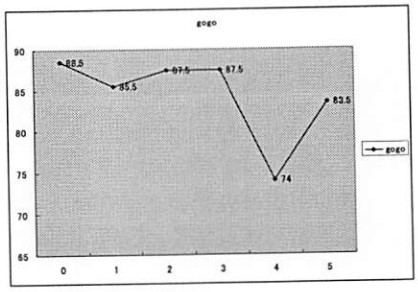


図 A.120: Fuzzy gogo

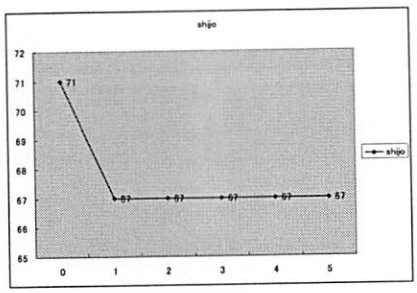


図 A.121: Fuzzy shijo

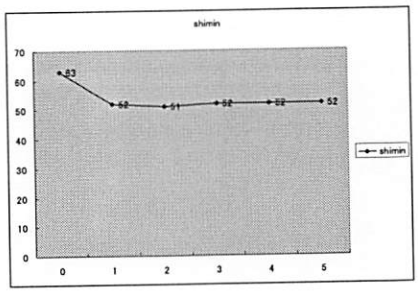


図 A.122: Fuzzy shimin

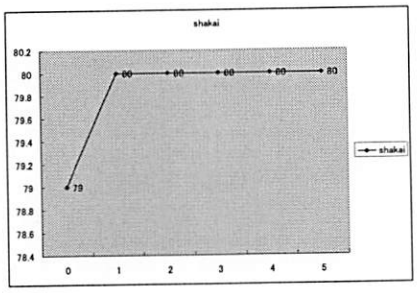


図 A.123: Fuzzy shakai

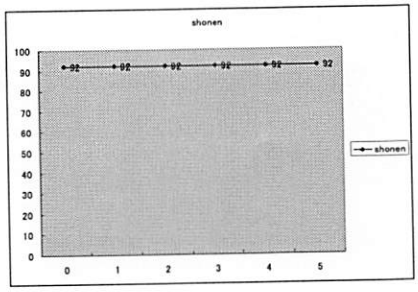


図 A.124: Fuzzy shonen

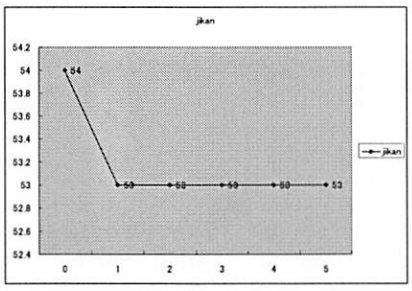


図 A.125: Fuzzy jikan

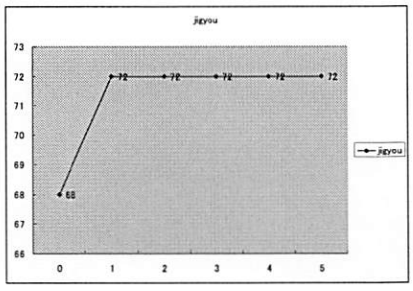


図 A.126: Fuzzy jigyou

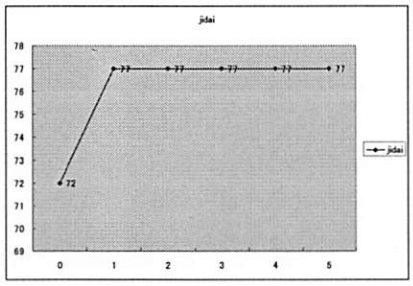


図 A.127: Fuzzy jidai

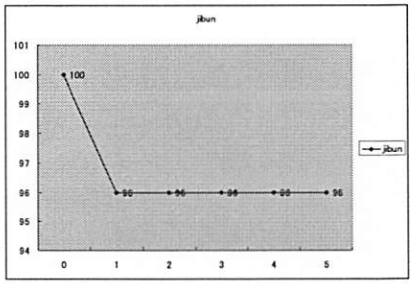


図 A.128: Fuzzy jibun

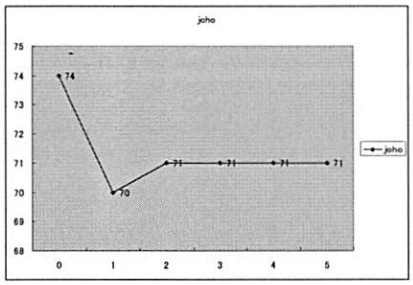


図 A.129: Fuzzy joho

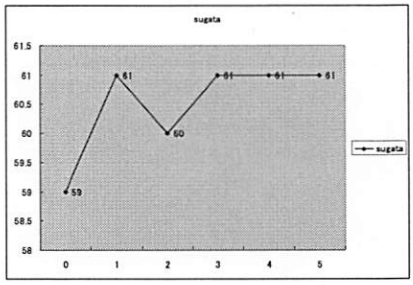


図 A.130: Fuzzy sugata

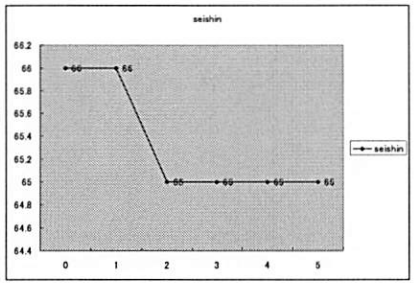


図 A.131: Fuzzy seishin

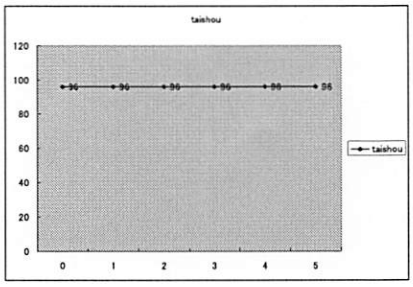


図 A.132: Fuzzy taishou

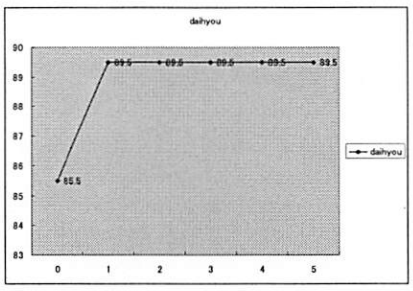


図 A.133: Fuzzy daihyou

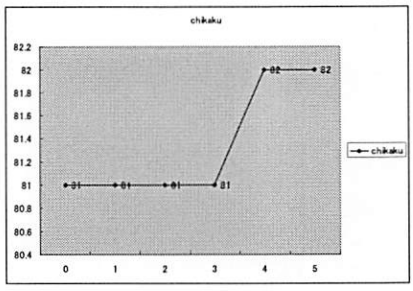


図 A.134: Fuzzy chikaku

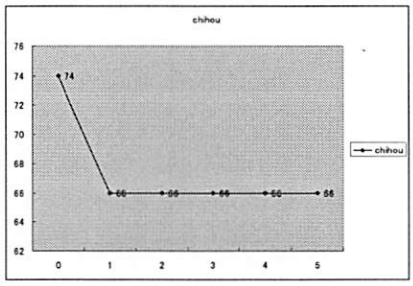


図 A.135: Fuzzy chihou

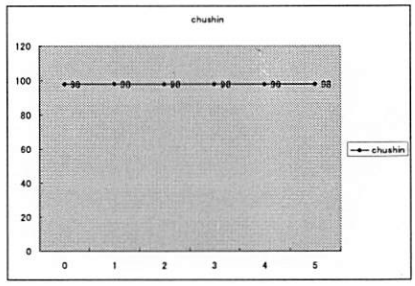


図 A.136: Fuzzy chushin

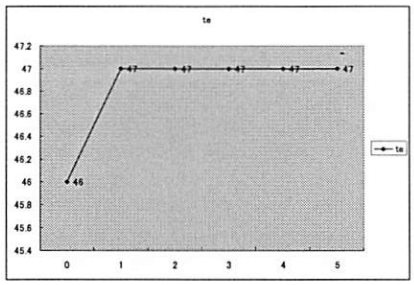


図 A.137: Fuzzy te

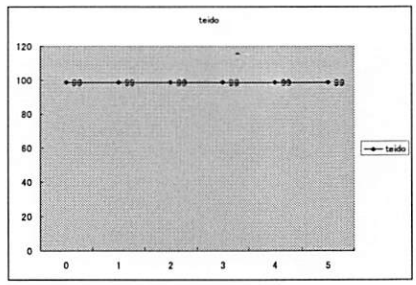


図 A.138: Fuzzy teido

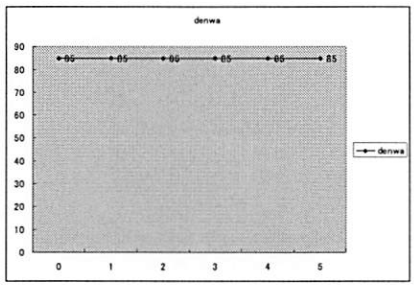


図 A.139: Fuzzy denwa

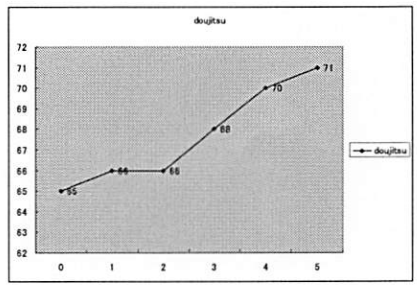


図 A.140: Fuzzy doujitsu

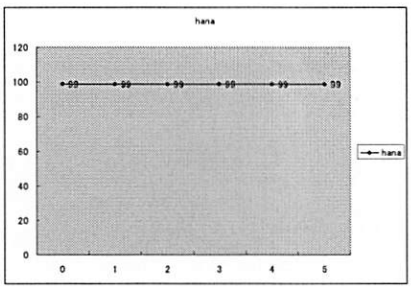


図 A.141: Fuzzy hana

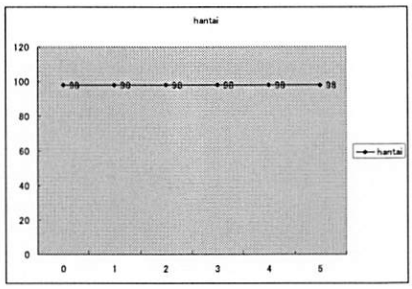


図 A.142: Fuzzy hantai

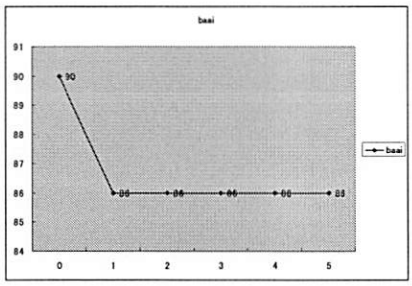


図 A.143: Fuzzy baai

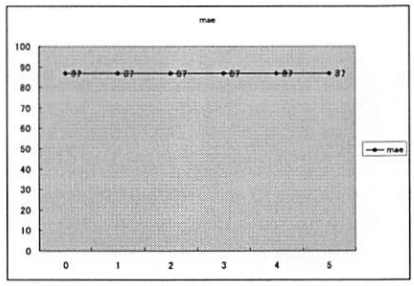


図 A.144: Fuzzy mae

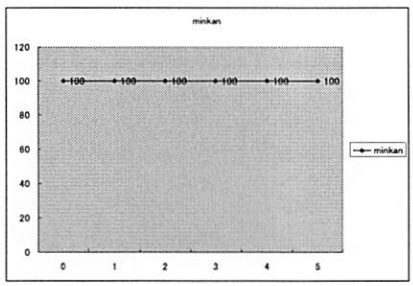


図 A.145: Fuzzy minkan

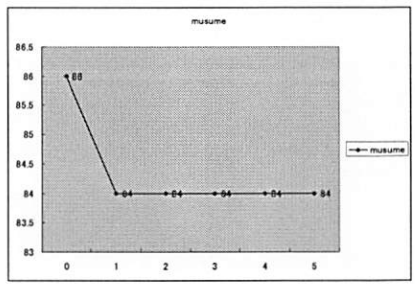


図 A.146: Fuzzy musume

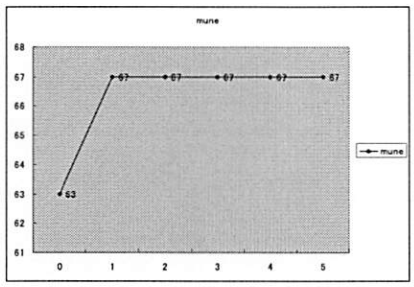


図 A.147: Fuzzy mune

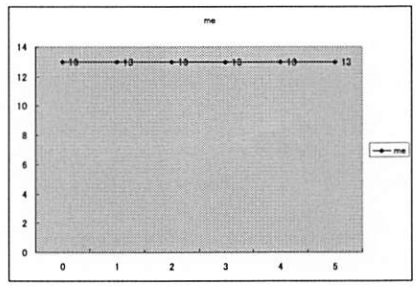


図 A.148: Fuzzy me

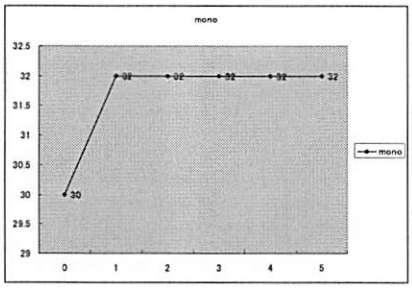


図 A.149: Fuzzy mono

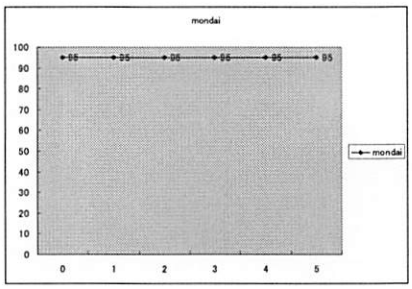


図 A.150: Fuzzy mondai

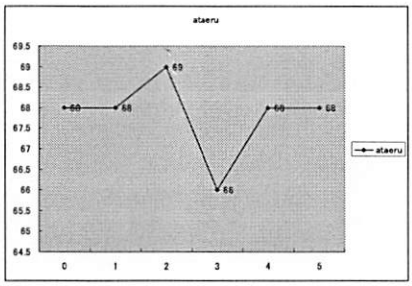


図 A.151: EM ataeru

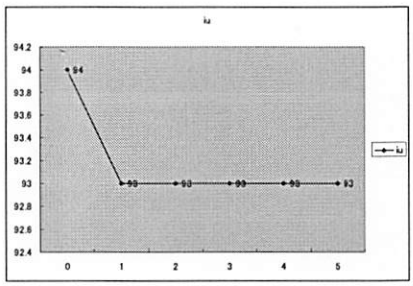


図 A.152: Fuzzy iu

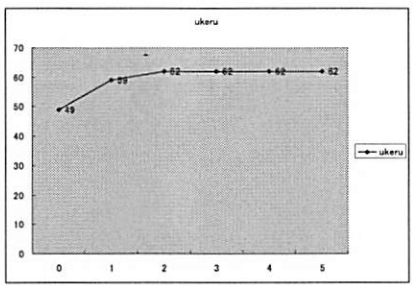


図 A.153: Fuzzy ukeru

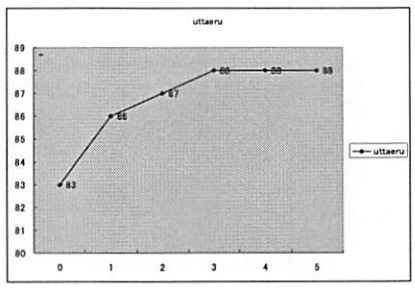


図 A.154: Fuzzy uttaeru

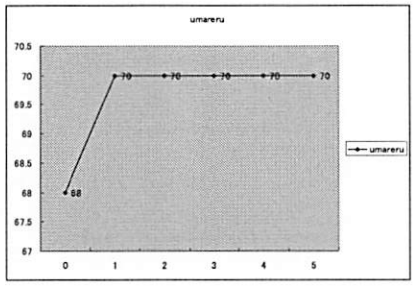


図 A.155: Fuzzy umareru

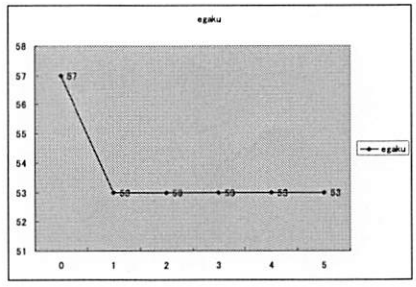


図 A.156: Fuzzy egaku

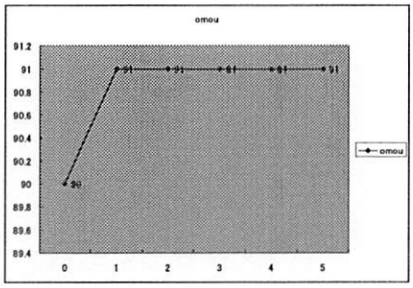


図 A.157: Fuzzy omou

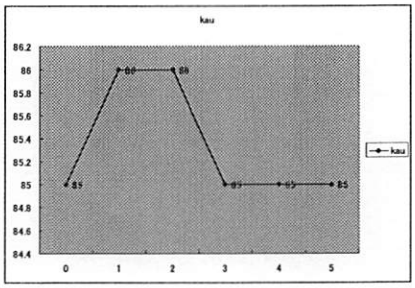


図 A.158: Fuzzy kau

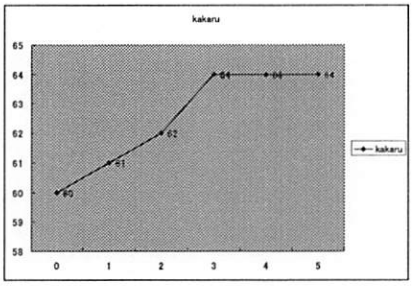


図 A.159: Fuzzy kakaru

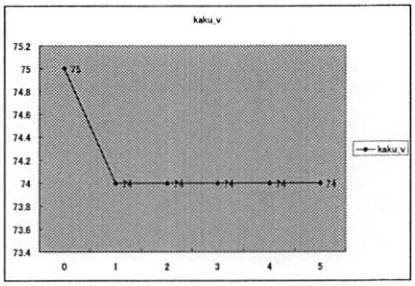


図 A.160: Fuzzy kaku_v

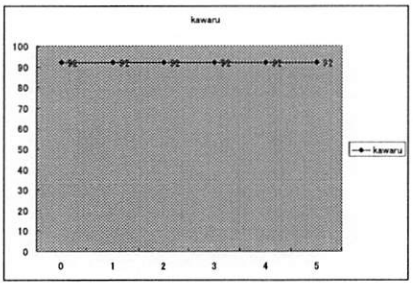


図 A.161: Fuzzy kawaru

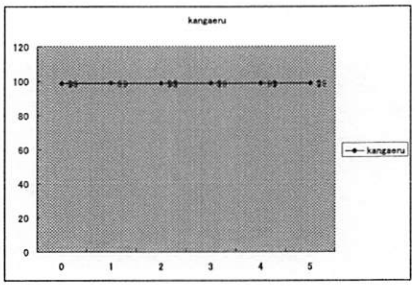


図 A.162: Fuzzy kangaeru

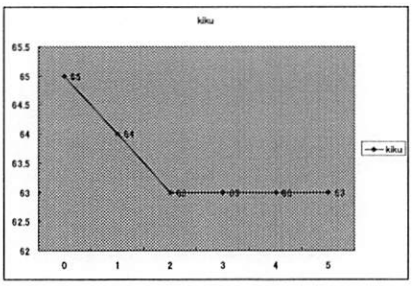


図 A.163: Fuzzy kiku

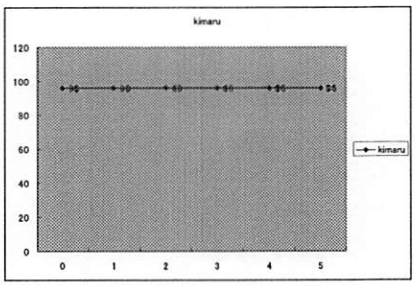


図 A.164: Fuzzy kimaru

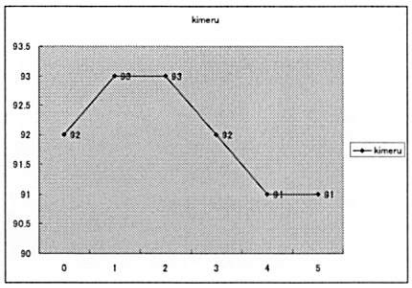


図 A.165: Fuzzy kimeru

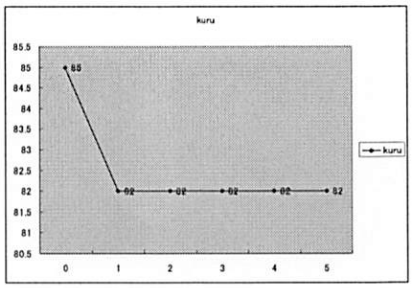


図 A.166: Fuzzy kuru

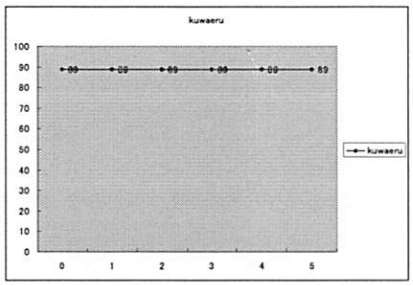


図 A.167: Fuzzy kuwaeru

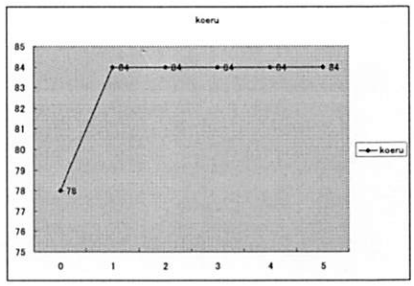


図 A.168: Fuzzy koeru

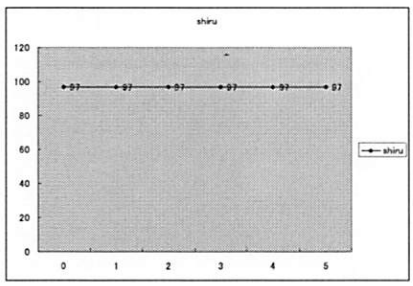


図 A.169: Fuzzy shiru

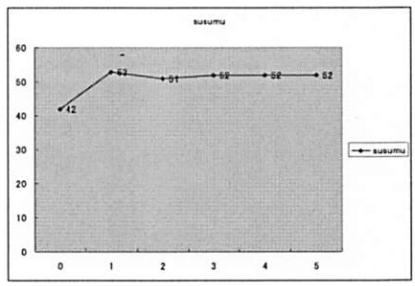


図 A.170: Fuzzy susumu

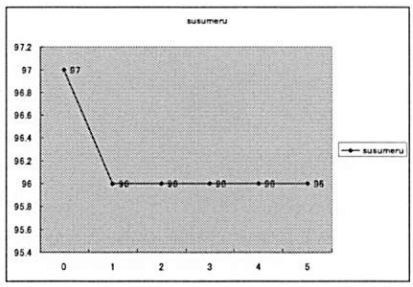


図 A.171: Fuzzy susumeru

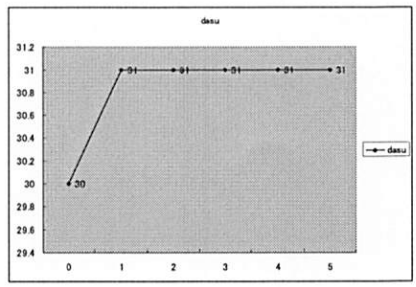


図 A.172: Fuzzy dasu

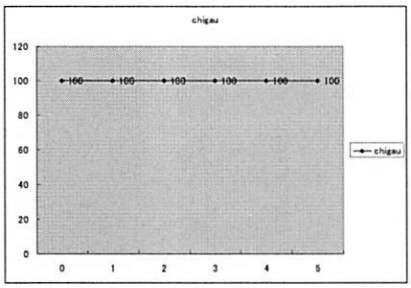


図 A.173: Fuzzy chigau

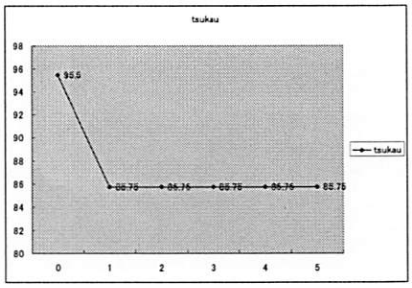


図 A.174: Fuzzy tsukau

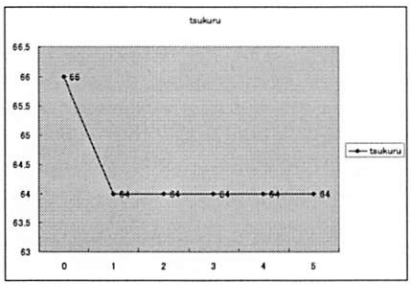


図 A.175: Fuzzy tsukuru

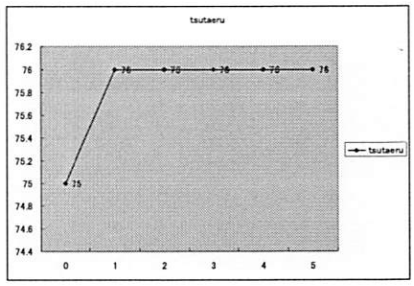


図 A.176: Fuzzy tsutaeru

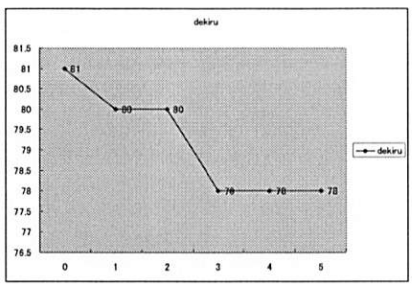


図 A.177: Fuzzy dekiru

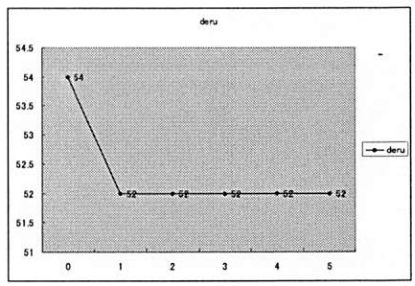


図 A.178: Fuzzy deru

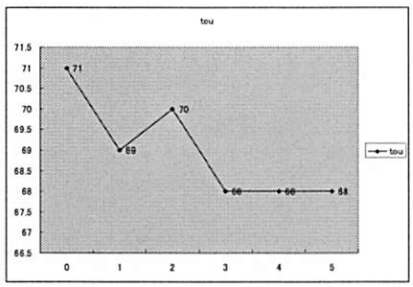


図 A.179: Fuzzy tou

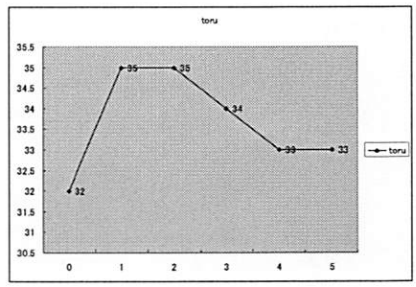


図 A.180: Fuzzy toru

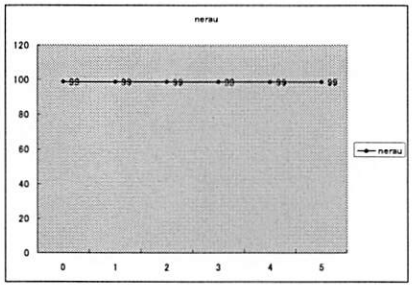


図 A.181: Fuzzy nerau

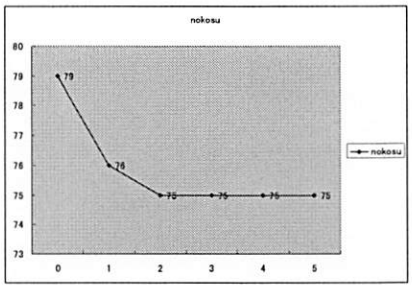


図 A.182: Fuzzy nokosu

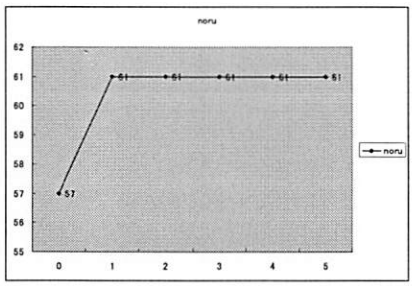


図 A.183: Fuzzy noru

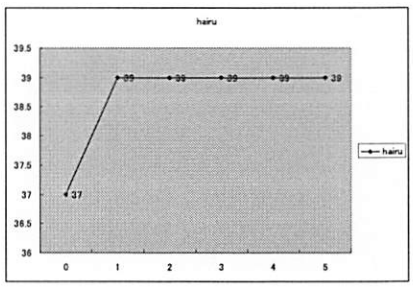


図 A.184: Fuzzy hairu

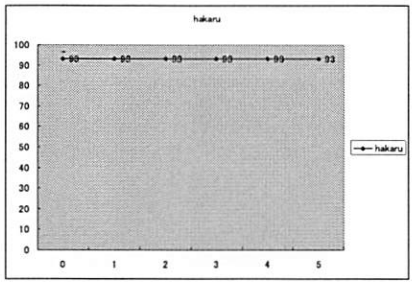


図 A.185: Fuzzy hakaru

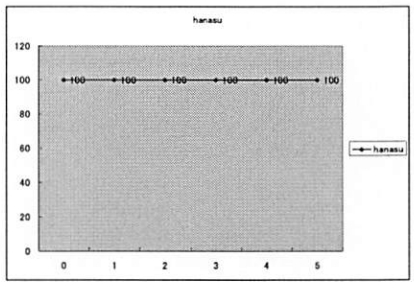


図 A.186: Fuzzy hanasu

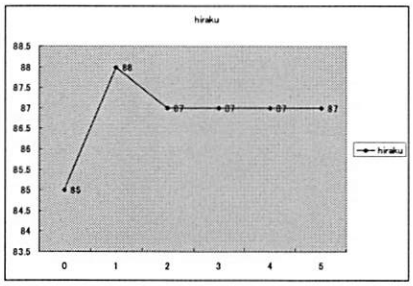


図 A.187: Fuzzy hiraku

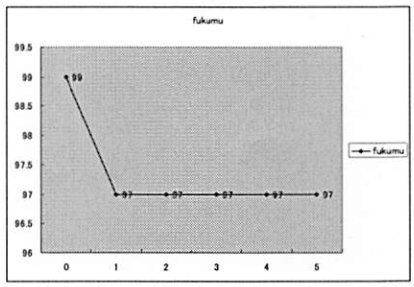


図 A.188: Fuzzy fukumu

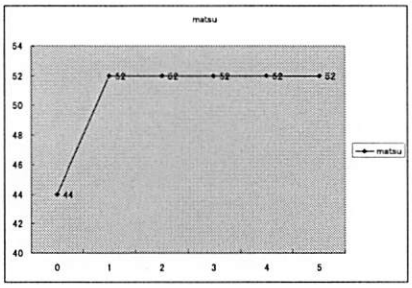


図 A.189: Fuzzy matsu

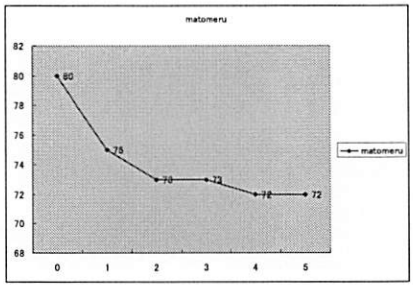


図 A.190: Fuzzy matomeru

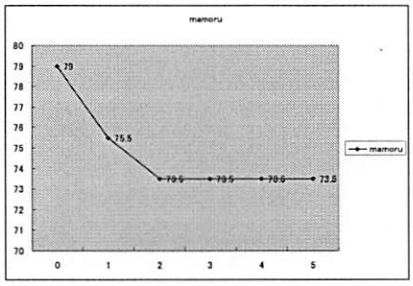


図 A.191: Fuzzy mamoru

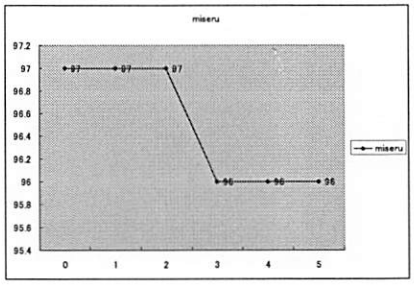


図 A.192: Fuzzy miseru

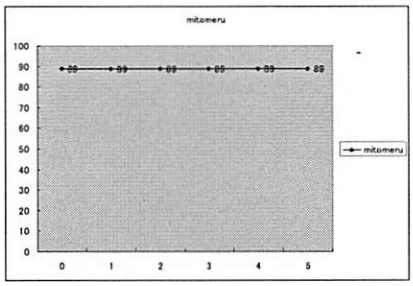


図 A.193: Fuzzy mitomeru

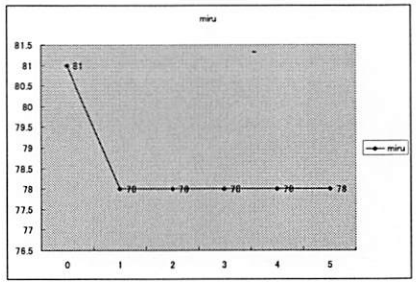


図 A.194: Fuzzy miru

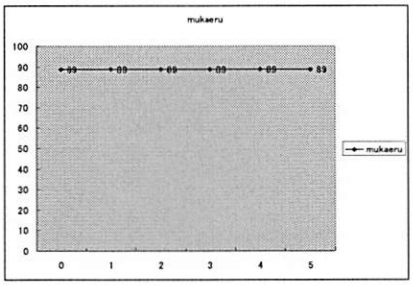


図 A.195: Fuzzy mukaeru

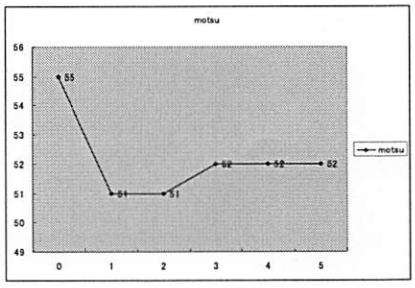


図 A.196: Fuzzy motsu

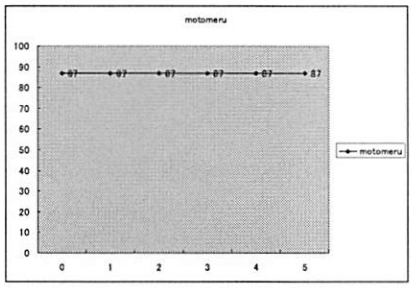


図 A.197: Fuzzy motomeru

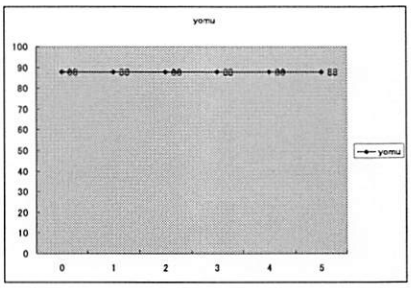


図 A.198: Fuzzy yomu

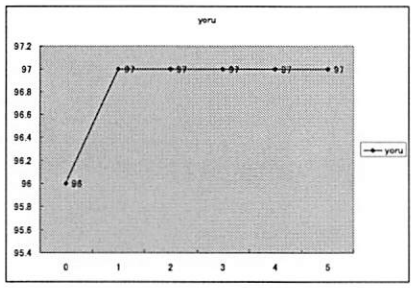


図 A.199: Fuzzy yoru

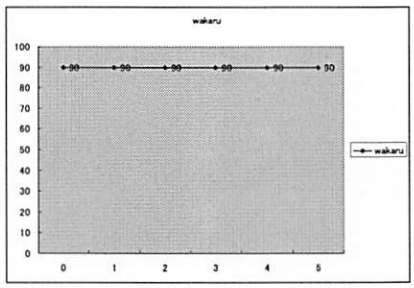


図 A.200: Fuzzy wakaru

# 付 録B　プログラムソースリスト（EMアルゴリズム）

```
#include<stdio.h>
#include<string.h>

/*式 (2.10) を計算*/
/*実行例--->a.out ev2id.dat all.dat expression6.txt*/

int get_class_prob(int num,char ab);

FILE *fin1,*fin2,*fout;

main(int argc,char *argv[]){

  int nums[2000];
  char data[2000],w[20],n[10];
  int i,j,k,pa,pb,v=6713,;
  int bunshia=0,bunshib=0;

  if(argc!=5){
    printf("引数の数が違う！\n");
    exit(1);
  }

  if((fin1=fopen(argv[1],"r")) == NULL){
    printf("f1のファイルが開けられない\n");
    exit(1);
```

```
  }

  if((fin2=fopen(argv[2],"r")) == NULL){
    printf("f2のファイルが開けられない\n");
    exit(1);
  }

  if((fout=fopen(argv[4],"a")) == NULL){
    printf(ファイルが開けられない\n);
    exit(1);
  }

  while(fgets(data,2000,fin1) != NULL){

    for(i=0;data[i]!=9;i++){
      w[i]=data[i];
    }
    i+=2;

    for(j=0,k=0;data[i]!='\0';i++){
      if(data[i]==9){
n[k]='\0';
if(atoi(n)<=100){
  nums[j]=atoi(n);
  j++;
}
k=0;i++;
      }
      n[k]=nums[i];
      k++;
    }
    nums[j]=0;
```

```
    for(i=0;nums[i]!=0;i++){
      pa=get_class_prob(nums[i],a);
      pb=get_class_prob(nums[i],b);

      bunshia+=pa;
      bunshib+=pb;
    }



  }

  fclose(fin1);
  fclose(fin2);
  fclose(fout);
}

int get_class_prob(int num,char ab){

  char d[256];
  int i=1,j=0;

  fseek(fin2,0L,SEEK_SET);

  while(fgets(d,256,fin2) != NULL){

    if(i==num){
      if(ab==97){
while(d[j]!='\0') j++;
if((d[j-1]==49)&&(d[j-2]==9)) return 1;
else return 0;
      }
      else if(ab==98){
```

```
while(d[j]!='\0') j++;
if((d[j-1]==50)&&(d[j-2]==9)) return 1;
else return 0;
      }
    }

    i++;
  }

}
```

```
#include<stdio.h>
#include<string.h>

/*P(c_j) を計算!!*/
/*実行例---> a.out labeled_100.txt*/

main(int argc,char *argv[]){


  FILE *fin;

  char ab[]="ab",data[256];
  float s=0;
  int i=0,j=0,d=0,p=0,sum_p=0;


  if(argc!=2){
    printf("引数の数が違う！\n");
    exit(1);
  }

  if((fin=fopen(argv[1],"r")) == NULL){
    printf("f1 ファイルが開けられない\n");
    exit(1);
  }


  for(i=0;i<2;i++){

    d=0;sum_p=0;

    fseek(fin,0L,SEEK_SET);
```

```
    while(fgets(data,256,fin)!=NULL){
      p=0;

      if(strchr(data,ab[i])!=NULL) {p=1;/*printf("p=1!!\n");*/}

      sum_p+=p;
      d++;
    }

    s=((1.0+sum_p)/(2.0+d));

    printf("%c:---s:%f\n",ab[i],s);

  }

  fclose(fin);

}
```

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

/*式 (2.11) を計算!!*/

char classify(char *e1,char *e2,char *e3,char *e4,char *e5,char *e6);


char classify(char *e1,char *e2,char *e3,char *e4,char *e5,char *e6){

  FILE *fin;
  char data[256],p_s[256],d[256];
  int i,j,k,e_num,sp=0;
  float bunbo,bunsi[2],p_ab[]={0.589,0.412};
  long p_n,p_sum=1.0;

  if((fin=fopen("expression6_100.txt","r")) == NULL){
    printf("入力ファイルがオープンできない。\n");
    exit(1);
  }

  //printf("IN FUNC_8!\n");

  for(i=0;i<=1;i++){

    fseek(fin,0L,SEEK_SET);

    while(fgets(data,256,fin)!=NULL){
      k=0;

      for(j=4;((data[j]!=97)&&(data[j]!=98));j++){
d[k]=data[j];
```

```
k++;
        }

        d[k-1]='\n';
        d[k]='\0';

        k=0;
        j+=2;
        while(data[j]!='\0'){
p_s[k]=data[j];
j++;k++;
        }
        p_s[k-1]='\n';
        p_s[k]='\0';

        puts(d);
        printf("\n");
        puts(p_s);
        //printf("\n");

        p_n=atof(p_s);
        printf("%f\n",p_n);

        if((data[1]==1)&&(strcmp(d,e1)==0)){
puts(d);
printf("\n");
p_sum*=p_n;
        }

        else if((data[1]==2)&&(strcmp(d,e2)==0)){
puts(d);
printf("\n");
p_sum*=p_n;
```

```
      }

      else if((data[1]==3)&&(strcmp(d,e3)==0)){
puts(d);
printf("\n");
p_sum*=p_n;
      }

      else if((data[1]==4)&&(strcmp(d,e4)==0)){
puts(d);
printf("\n");
p_sum*=p_n;
      }

      else if((data[1]==5)&&(strcmp(d,e5)==0)){
puts(d);
printf("\n");
p_sum*=p_n;
      }

      else if((data[1]==6)&&(strcmp(d,e6)==0)){
puts(d);
printf("\n");
p_sum*=p_n;
      }

    }

    bunsi[i]=p_ab[i]*p_sum;
  }

  bunbo=bunsi[0]+bunsi[1];
```

```
  if((bunsi[0]/bunbo)>(bunsi[1]/bunbo)) return 'a';
  else if((bunsi[0]/bunbo)<(bunsi[1]/bunbo)) return 'b';
  else return 'c';
}
```

# 付 録C　プログラムソースリスト（ファジィクラスタリング）

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<math.h>

double cal_d(int,int);
double cal_u(int,int);
double cal_v(int,int);
double cal_diff(int);

FILE *fin,*fin2,*fout;
int N=0,C=0,P=0,n=0;

////////////////////////////
//double u[C][N],v[C][P];///
//double x[N][P],vn[C][P];//
////////////////////////////

double u[8][12000],v[8][9000];
float x[9000][9000];
//double xg[12000][60];
//int yg[12000][60];
double vn[8][9000];
//double x2[12000];
double xx[12000];
```

```
double vv[15];
//double u[3][6],v[3][6];
//double x[6][6];
//double vn[3][6];
int x_freq[12000];

main(int argc,char *argv[]){

  int i=0,ii=0,k=0,g=0,m=0,J=0,p=0,q=0;
  int flag=0,count=0,check=0,s;
  double max=0.0,diff=0.0;
  char num[10],x_num[10];
  //char s[256],num[10];
  char word[15000][64],w[64];
  //char word[6][64],w[64];

  if(argc!=4){
    printf("引数の数が違います。\n");
    exit(1);
  }

  if((fin=fopen(argv[1],"r")) == NULL){
    printf("arg[1], ファイルが開けられません。\n");
    exit(1);
  }

  if((fin2=fopen(argv[2],"r")) == NULL){
    printf("arg[2], ファイルが開けられません。\n");
    exit(1);
  }

  if((fout=fopen(argv[3],"w")) == NULL){
    printf("arg[3], ファイルが開けられません。\n");
```

```
    exit(1);
  }

  ////////////////////wordおよびN,C,Pを取得//////////////////////
  ////////////////////////数値代入////////////////////////////

  while((s=getc(fin)) != EOF){
    if(s != '\n'){
      //if(s == '\0') break;
      if((s != ':')&&(s != ' ')){
if(check==0){
  w[q]=s;
  //printf("C:%d  w[%d]:%d\n",C,q,w[q]);
  q++;
}
if(check==1){
  num[q]=s;
  printf("C:%d  num[%d]:%d\n",C,q,num[q]);
  q++;
}
      }
      if(s == ':'){
//printf("s=:\n");
w[q]='\0';q=0;
printf("w:{%s}\n",w);
check=1;
      }
      if(s == ' '){
num[q]='\0';q=0;
printf("num:{%s}\n",num);
check=0;

for(g=0;g<P;g++){
```

```
  if(strcmp(word[g],w)==0){
    printf("strcmp(word[g],w)==0\n");
    vn[C][g]=atof(num);
    printf("vn[%d][%d]:%f\n",C,g,vn[C][g]);
    flag=1;
    break;
  }
}
printf("x\n");
if(flag == 0){
  printf("xx\n");
  strcpy(word[P],w);
  printf("word[%d]:%s\n",P,word[P]);
  printf("num:%s\n",num);
  vn[C][P]=atof(num);
  printf("vn[%d][%d]:%f\n",C,P,vn[C][P]);
  P++;printf("P++\n");
}
else flag=0;
      }


    }
    else {
    printf("C++\n");
    C++;
    }
  }
  q=0;check=-1;

  while((s=fgetc(fin2)) != EOF){
    if(s != '\n'){
      //if(s == '\0') break;
      if((s != ':')&&(s != ' ')){
```

```
if(check==0){
  w[q]=s;
  q++;
}
if(check==1){
  num[q]=s;
  q++;
}
if(check==-1){
  x_num[q]=s;
  q++;

}
      }
      else if(s == ':'){
w[q]='\0';q=0;
check=1;
      }
      else if(s == ' '){
if(check==-1){
  x_num[q]='\0';q=0;
  check=0;
}
       if(check==1){
  num[q]='\0';q=0;
  check=0;

  for(g=0;g<P;g++){
    if(strcmp(word[g],w)==0){
      x[n][g]=atof(num);
      flag=1;
      break;
    }
```

```
  }
  if(flag == 0){
    strcpy(word[P],w);
    x[n][P]=atof(num);
    P++;
  }
  else flag=0;
}
      }
    }
    else{
      //printf("N:%d\n",N);
      if(0 >= (x_freq[n]=atoi(x_num))){
//printf("x_freq[%d]=<0.....%d\n",n,x_freq[n]);
u[((-1)*(x_freq[n]))][n]=1.0;
x_freq[n]=1;
      }
      //printf("x_freq[%d]:%d\n",n,x_freq[n]);
      N+=x_freq[n];
      n++;check=-1;
    }
  }
  printf("P:%d,N:%d,n:%d,C:%d \n",P,N,n,C);
  printf("keisan!\n");
  ////////////////////////計算////////////////////////////

  for(i=0;i<C;i++){
    for(g=0;g<P;g++){
      v[i][g]=vn[i][g];
      vv[i]+=pow(vn[i][g],2);
      //    printf("vn[%d][%d]:%f\n",i,g,vn[i][g]);
    }
  }
```

```
  for(k=0;k<n;k++){
    for(g=0;g<P;g++){
    //for(m=0;yg[k][m] != -1;m++){
      //x2[k]+=x[k][g];
      xx[k]+=pow(x[k][g],2);
      //if(m>=60) break;
      //printf("x[%d]:%f\n",k,x[k]);
      //printf("x2[%d]:%f\n",k,x2[k]);
    }
  }

  while(1){
    for(k=0;k<n;k++){
      for(i=0;i<C;i++){
if(u[i][k]==1.0){
  printf("u[%d][%d]==1.0\n",i,k);
  for(ii=0;ii<C;ii++){
    if(ii==i) {continue;}
    u[ii][k]=0.0;
    printf("u[%d][%d]:%f\n",ii,k,u[ii][k]);
  }
  //u[i][k]=1.0;
  break;
}
else{
  u[i][k]=cal_u(k,i); ///////////FCM 2.
  printf("u[%d][%d]:%f\n",i,k,u[i][k]);
}
//J+=u[i][k]*u[i][k]*cal_d(k,i);
      }
    }
    ////////////////////////////////////u
    //for(i=0;i<C;i++){
```

```
    //  printf("u[%d] : ",i);
    // for(k=0;k<N;k++){
    //  printf(" %f ", u[i][k]);
    // }
    // printf("\n");
    // }
    ////////////////////////////////////

    for(i=0;i<C;i++){
      for(g=0;g<P;g++){
vn[i][g]=cal_v(i,g); //////////FCM 3.
//printf("vn[%d][%d]:%f\n",i,g,vn[i][g]);
      }
      diff=cal_diff(i);
      //printf("diff:%f\n",diff);
      if(max < diff) max=diff;
    }

    ////////////////////////////////////v_
    // for(i=0;i<C;i++){
    // printf("v_[%d] : ",i);
    // for(g=0;g<P;g++){
    //  printf(" %f ",vn[i][g]);
    // }
    // printf("\n");
    // }
    ////////////////////////////////////

    if(count>=0) {break;}
    //printf("count : %d\n",count);
    //if(max < 0.0001) {break;}   //////////FCM 4.
    //if(max < 0.1) {break;}   //////////FCM 4.
    else{
```

```
      for(i=0;i<C;i++){
vv[i]=0.0;
for(g=0;g<P;g++){
  v[i][g]=vn[i][g];
  vv[i]+=pow(vn[i][g],2);;
}
      }
    }
    max=0.0;
    diff=0.0;
    printf("%d end!\n",count);
    count++;
  }

  printf("hyouji!\n");
  ////////////////////////表示////////////////////////
  for(i=0;i<C;i++){
    for(g=0;g<P;g++){
      fprintf(fout,"%s:%f ",word[g],vn[i][g]);
    }
    fprintf(fout,"\n");
  }


  fclose(fin);
  fclose(fin2);
  fclose(fout);

}

////////////////////////関数////////////////////////////
```

```
double cal_d(int k,int i){
  int g=0,m=0;
  double d2=0.0;

  for(g=0;g<P;g++){
    //d1+=pow(vn[i][g],2);
    if(vn[i][g]!=0) d2+=((-2)*x[k][g]*vn[i][g]);
    //d+=((get_x(k,g)-vn[i][g])*(get_x(k,g)-vn[i][g]));
    //d+=((x[k][g]-vn[i][g])*(x[k][g]-vn[i][g]));
    //}
    //for(m=0;yg[k][m] != -1;m++){
    //printf("yg[%d][%d]=%d\n",k,m,yg[k][m]);
  }
  return (xx[k]+vv[i]+d2);
}

double cal_u(int k,int i){
  int j=0;
  double uu=0.0;
  double bunshi=0.0;

  bunshi=cal_d(k,i);
  //if(bunshi==0) return(1.0);
  //else{
  for(j=0,uu=0.0;j<C;j++){
    uu+=(bunshi/cal_d(k,j));
  }
  return(1/uu);
  //}
}

double cal_v(int i,int g){
  double bunbo=0.0,bunsi=0.0;
```

```
  int k,l=0;

  for(k=0;k<n;k++){
    bunbo+=(x_freq[k]*(u[i][k]*u[i][k]));
    //bunsi+=(u[i][k]*u[i][k]*get_x(k,g));
    bunsi+=(x_freq[k]*(u[i][k]*u[i][k]*x[k][g]));
  }
  return(bunsi/bunbo);
}

double cal_diff(int i){
  int g=0;
  double diff=0.0;

  for(g=0;g<P;g++){
    diff+=((vn[i][g]-v[i][g])*(vn[i][g]-v[i][g]));
  }

  return(sqrt(diff));
}

/*
double get_x(int k,int g){
  int m=0;

  //for(m=0;m<7000;m++){
  for(m=0;yg[k][m] != -1;m++){
    if(yg[k][m]==g) return(xg[k][m]);
    if(m>=60) break;
  }
  return(0.0);
}
*/
```