平成２０年度茨城大学工学部情報工学科卒業研究論文

次元縮約による混合分布モデルを利用した文書クラスタリング

平成21年２月１０日

茨城大学工学部情報工学科
田中　洸一（05T4047H）
新納　浩幸　准教授

平成２０年度　茨城大学工学部情報工学科　卒業研究

次元縮約による混合分布モデルを利用した文書クラスタリング

著者：田中洸一（05T4047H）
指導教員：新納浩幸　准教授

論文趣旨

インターネットの普及や社会の情報化に伴い、大量の情報が文書という形で存在する。情報を有効活用するために、文書データを整理管理していくことが重要になってきている。文書クラスタリングはそのための基本要素技術であり、文書の集合をその内容の類似性からグループ分けする技術である。

文書クラスタリングの手法は多岐にわたるが、近年、確率統計的な枠組みをもった混合分布モデルに基づく手法が提案されている。この手法は理論的な背景をもち、理論的かつ頑健なクラスタリングが可能である。

ただし混合分布モデルでは計算量が膨大であり、扱うデータが比較的低次元のベクトルでないと、実際には処理が不可能である。文書クラスタリングの場合、データにあたる文書は一般にはベクトル空間モデルによりベクトル化されるために、非常に高次元(１万次元以上)であり、混合分布モデルの利用は困難である。ここでは次元縮約の手法を利用して文書のベクトルを低次元に変換することで混合分布モデルの利用を試みる。

次元縮約の手法としては、SVD、PLSI、NMF の３つの手法がある。どの手法で次元縮約すれば混合分布モデルのクラスタリングを利用できるかを検討する。実験ではSVD、PLSI、NMF による次元縮約を行ったベクトルに対して混合分布モデルを用いたクラスタリングと標準的なクラスタリング手法の k-means との比較を行った。

精度的には k-means の方が優れていた。その原因は次元縮約を行った際に、混合分布モデルで仮定されている各次元の分布の正規性が破られているためだと予想し、各次元の正規性の検定を行い、この予想を確認した。

目次

第１章　はじめに

１．１　背景と目的

さまざまな文書データが取り扱われている昨今、これからもますますその扱う情報は増え続けるだろう。そういった中で、それらの情報を整理し管理していくことが重要になってきている。膨大な文書データをあるカテゴリーごとに分別するシステムがあれば望ましい。ひとつの案として、文書クラスタリングがある。

文書クラスタリングの手法は多岐にわたるが、近年、確率統計的な枠組みをもった混合分布モデルに基づく手法が提案されている。この手法は理論的な背景をもち、理論的かつ頑健なクラスタリングが可能である。

ただし、混合分布モデルでは、扱うデータが比較的低次元のベクトルでないと処理ができない。文書データはベクトル空間モデルによりベクトル化されるため、非常に高次元なベクトルになってしまい、計算量が膨大になる。

そこで、次元縮約の手法を用いて混合分布モデルの利用を試みる。手法としては「SVD」「pLSI」「NMF」の３つの手法がある。本研究ではSVD、pLSI、NMFによる次元縮約によって混合分布モデルでの文書クラスタリングを行い、有効性を検証するために、標準的なクラスタリング手法の「kmeans」と比較する。

１．２ 本論文の構成

第２章でクラスタリングの基本と文書データの扱い方について述べる。ここでクラスタリングの比較対象であるk-meansについてふれる。第３章では、混合分布モデルについて述べる。第４章では、次元縮約について述べる。第５章で実験結果、第６章で考察、第７章でまとめ、このように構成されている。

第２章　クラスタリング手法

２．１　文書クラスタリングとは

クラスタリングとは、データ集合から似たようなデータを集めグループ分けをすることである。文書クラスタリングは、文書の主題の類似性からグループ分けをする。

２．２　文書データの表現法

クラスタリングのために、文書データをn次元のベクトルで表現する必要がある。一般にベクトル空間モデルを使う。ベクトル空間モデルでは、文書集合中に現れた単語w_iごとの頻度を値として第i次元に設定する。

しかし、その表現方法だと文書中のすべての単語が、同じ重みで評価されてしまう。単語には、文書の主題を推定できるような単語もあれば、どんな文書にも出現する単語もある。前者には重みを重くし、後者には重みを軽く設定したほうが、よりその文書の特徴を表すことができ、クラスタリングしやすくなる。

重みのつけ方として、TF＊IDF が標準的である。TF は文書d_i中の単語w_jの頻度である。IDF は全文書数Nをw_jを含む文書数n_jで割った値に対数をとったものである。TF をf_{ij}と表記すると重みは、

$$f_{ij} * \log\left(N/n_j\right)$$

となる。さらに IDF が０になると不都合になる場合があるので、その場合は$\log\left((N+1)/n_j\right)$に補正することもある。

また、サイズの大きな文書は、大きなベクトルになってしまう。そのためにベクトルの長さを１に正規化する処理も行われる。

２．３　階層的手法と非階層的手法

クラスタリングの手法は、２つに分類される。階層的手法と非階層的手法である。階層的手法は、与えられたデータセットの各データが１つのクラスタとなっている状態を初期状態として、クラスタ間の距離や類似度に基づいて、２つのクラスタを逐次的に併合していく手法である。目的のクラスタ数、または１つのクラスタになるまで併合を繰り返す。

非階層的手法は、データの分割の良さを表すある評価関数を設定し、その評価関数に対する最適解を探索することでクラスタリングを行う手法である。以下にまとめる。

階層的手法	単純結合、完全連結法、群平均法、ウォード法、重心法、メディアン法
非階層的手法	k-means、混合分布モデル、スペクトラルクラスタリング、PLSI、NMF、Fuzzy c-means

今回、実験で扱うクラスタリング手法は、非階層的手法の「k-means」と「混合分布モデル」である。「k-means」については、以下に、「混合分布モデル」は第三章で紹介する。

２．４　k-means

k-means は、分割後のクラスタの数Kをあらかじめ与える必要がある。よって、データセットとKを入力し、以下の手順で実行する、

１、　K個のクラスタの代表点$c_1, c_2, \cdots c_K$をデータセットから適当に選ぶ。
２、　各データxに対して、xとc_iとの距離を測り、最も距離の短いc_iに対するクラスタをxのクラスタに設定する。
３、　２により各データxのクラスタが変わらなければ終える。変わるなら、各クラスタの重心を代表点$c_1, c_2, \cdots c_K$に設定し、２に戻る。

つまり、k-means はクラスタの重心をそのクラスタの代表点$c_1, c_2, \cdots c_K$に設定して、次の評価関数が最小化するようなデータのクラスタへの割り当てを求めることである。

$$\sum_{i=1}^{K} \sum_{x \in C_i} \|x - c_i\|^2$$

アルゴリズム（１～３）によって式の値は単調に減少していく。しかし、そのアルゴリズムによって得られた値は大域解ではなく局所解である。初期値の選び方によって、得られる解が異なる。初期値を変化させ、数回実行し、得られた解の中からもっともよい解を選んだほうがよい。

２．５　クラスタリングの評価法

クラスタリング結果を評価するには、正解集合が用意されているかどうか、クラスタの数が与えられているかどうかによって、様々な方法がある。ただしどの評価方法にしても長所と短所があり、標準的な評価方法は確立されていない。今回実験で扱う２つの評価方法（エントロピー、純度）を紹介する。

２．５．１ エントロピー（entropy）

最も標準的に用いられているクラスタリングの評価尺度はエントロピーである。各クラスタ C_i に対するエントロピー E_i を求めて、クラスタのデータ後による重み付き平均をとることで全体のエントロピーが定義される。つまり、以下の式がクラスタリング結果のエントロピーである。この値は０から１の値をとり、値が低いほどクラスタリング結果が良好であることを意味する。

$$\sum_{i=1}^{K}\frac{|C_i|}{N}E_i=\sum_{i=1}^{K}\frac{\sum_{j=1}^{K}x_{ij}}{N}E_i$$

ここで、N はクラスタリング対象のデータ数を表す。また、E_i は以下で定義される。

$$E_i=-\sum_{h=1}^{K}P(A_h\mid C_i)\log P(A_h\mid C_i)\quad(2.1)$$

ここで確率 $P(A_h\mid C_i)$ が出てくるが、これは、

$$\frac{|A_h\cap C_i|}{|C_i|}=\frac{x_{ih}}{\sum_{j=1}^{K}x_{ij}}$$

によって推定する。

また、式（２．１）には対数 log が出ている。対数の低はいくつでもかまわないが、クラスタ数　を使うことが多い。

２．５．２ 純度（purity）

エントロピーと同様、純度も標準的な評価尺度である。

クラスタ C_i に対する純度 P_i とは、ある正解のクラスタのデータをどの程度含むかという指標であり、以下で定義される。

$$P_i = \frac{1}{|C_i|} \max_h |C_i \cap A_h|$$

クラスタリング結果の純度は、各クラスタのデータ数による重み付き平均をとることで定義される。つまり、以下が定義である。

$$\sum_{i=1}^{K} \frac{|C_i|}{N} P_i = \frac{1}{N} \sum_{i=1}^{K} \max_h |C_i \cap A_h|$$

この値は０から１の値をとり、値が高いほどクラスタリング結果が良好であることを意味する。

第３章　混合分布モデル

３．１　混合分布モデルとは

混合分布モデルとは、複数の確率モデルである。クラスタリングの問題を確率モデルとしてモデル化して解く方法である。

データを発生する確率モデルがｋ個あるとする。c番目のモデルの下でデータxが発生する確率を$p_c(x)$とする。そして、データxが発生する確率$p(x)$をそれら確率の線形和で表現する。この確率の分布を混合分布とよぶ。

$$p(x)=\sum_{c=1}^{K}\alpha_c p_c(x)$$

データが連続値の場合は、確率$p(x)$と$p_c(x)$は確率密度関数$f(x)$と$f_c(x)$に置き換わって以下となる。

$$f(x)=\sum_{c=1}^{K}\alpha_c f_c(x)$$

ここで、$\alpha_c f_c(x)$はc番目のモデルとデータxが生起する同時確率に対応する。そこで、$\alpha_c f_c(x)$の中で最も大きな値をとる$c=\hat{c}$をデータxのクラスタ番号に割り当てることによりクラスタリングが行える。つまり各cに対するα_cと$f_c(x)$の具体的な式を作ることが目標となる。

$f_c(x)$は正規分布を仮定する。データxはx次元ベクトルなので、n変数の多変量正規分布である。

$$f_c(x)=\frac{1}{(2\pi)^{n/2}\left|\sum_c\right|^{1/2}}\exp\left\{-\frac{1}{2}(x-\mu_c)'\sum_c^{-1}(x-\mu_c)\right\}$$

μ_cはn次元の平均ベクトルである。

$$\mu_c=(\mu_1,\mu_2,\cdots,\mu_n)'$$

また、$\sum_c$ は$n \times n$の分散共分散行列である。

$$\sum_c = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \cdots & \sigma_{1n}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \cdots & \sigma_{2n}^2 \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{n1}^2 & \sigma_{n2}^2 & \cdots & \sigma_{nn}^2 \end{pmatrix}$$

$f_c(x)$を求めるには、パラメータμ_cと$\sum_c$ が求まればよい。このパラメータが各cについて存在する。求めるのはこれらのパラメータとα_cである。

観測されたデータ$x_1, x_2, \cdots, x_N$からそのデータを発生させる確率モデルのパラメータθを求めるには最尤法を使う。つまり、対数尤度関数$L(\theta)$を最大にするθを求めればよい。

$$L(\theta) = \sum_{i=1}^{N} \log \left\{ \sum_{c=1}^{K} \alpha_c f_c(x_i) \right\} \qquad (3.1)$$

しかし、式（３．１）が最大になるθを求めることはかなり困難である。そのため、EMアルゴリズムという方法を用いる。

３．２　EMアルゴリズム

EMアルゴリズムは式の最大化問題の準最適解を求める画期的な方法である。EMアルゴリズムは隠れ変数が存在し、その隠れ変数の値を得ることができれは、その他のパラメータが推定でき、しかもその他のパラメータが推定できれば、隠れ変数の値も推定できるような場合に使える。

混合分布の場合、各データxが何番目のクラスタから発生したかが分かると、パラメータの推定は各クラスタに属するデータだけを集めて行えばよい。そこで現実のデータを、c番目のクラスタから発生したという情報が欠けている不完全データとみなす。xに対する完全データを$y=(x,c)$とする。cはxが発生したクラスタの番号である。EMアルゴリズムはxの分布の最適化問題を、yの分布の最適化問題の繰り返し演算に帰着させる方法である。ここでyの分布を$q(y)$とおいておく。

EMアルゴリズムはE-stepとM-stepの２つのステップからなり、この２つのステップを収束するまで交互に繰り返す。今、ｔ回目の繰り返しで得られたパラメータの値を$\theta^{(t)}$とおく。E-stepではデータxと$\theta^{(t)}$の下での$\log q(y)$の平均を求め、それをQとおく。

$$Q=E\left[\log q(y|x,\theta^{(t)})\right]$$

M-stepでは、上のQを最大にするθを求め、それを$\theta^{(t+1)}$とする。E-stepで行っていることは、θを固定して、尤度を最大にする隠れ変数を求めることに対応する。また、M-stepはE-stepで得られた隠れ変数を固定して、尤度を最大にするθを求めることに対応する。

混合分布の場合、$y=(x,c)$の確率密度関数は以下のとおりである。

$$\alpha_c f_c(x)$$

よって、xと$\theta^{(t)}$が与えられたときcの分布は、

$$\frac{\alpha_c^{(t)} f_c^{(t)}(x)}{\sum_{k=1}^{K}\alpha_k^{(t)} f_k^{(t)}(x)}$$

となるので、Qの式は以下となる。

$$Q=\sum_{c=1}^{K}\frac{\alpha_c^{(t)} f_c^{(t)}(x)}{\sum_{k=1}^{K}\alpha_k^{(t)} f_k^{(t)}(x)}\log(\alpha_c f_c(x))$$

実際の観測データは複数個の$x_1, x_2, \cdots, x_N$だが、サンプルの独立性から各Qの和が全体のQとなる。

$$Q = \sum_{i=1}^{N} \sum_{c=1}^{K} \frac{\alpha_c^{(t)} f_c^{(t)}(x_i)}{\sum_{k=1}^{K} \alpha_k^{(t)} f_k^{(t)}(x_i)} \log(\alpha_c f_c(x_i))$$

$$= \sum_{i=1}^{N} \sum_{c=1}^{K} g_{ic}^{(t)} \log(\alpha_c f_c(x_i))$$

上記では、簡単化のために以下のようにおいた。

$$g_{ic}^{(t)} = \frac{\alpha_c^{(t)} f_c^{(t)}(x_i)}{\sum_{k=1}^{K} \alpha_k^{(t)} f_k^{(t)}(x_i)}$$

最終的に得られるg_{ic}がクラスタリング結果を表す。

次に、Qを最大にするθを求める。まず、α_cを求める。α_cには$\sum_{c=1}^{K} \alpha_c = 1$という条件が付いていたので、ラグランジェの未定乗数法を利用して、

$$Q + \lambda \left(\sum_{c=1}^{K} \alpha_c - 1 \right)$$

の極値問題を解く。α_cで上記の式を偏微分して、以下が得られる。

$$\frac{\sum_{i=1}^{N} g_{ic}^{(t)}}{\alpha_c} + \lambda = 0$$

ここから、

$$\lambda \alpha_c = -\sum_{i=1}^{N} g_{ic}^{(t)}$$

なので、両辺cによる総和をとれば、

$$\lambda = -\sum_{c=1}^{K} \sum_{i=1}^{N} g_{ic}^{(t)} = -\sum_{i=1}^{N} \sum_{c=1}^{K} g_{ic}^{(t)} = -N$$

となり、以下のようになる。

$$\alpha_c = \frac{1}{N} \sum_{i=1}^{N} g_{ic}^{(t)}$$

これが $\alpha_c^{(t+1)}$ となる。

次に、μ_c と Σ_c を求める。これは、

$$Q = \sum_{i=1}^{N} \sum_{c=1}^{K} g_{ic}^{(t)} \log(\alpha_c^{(t+1)} f_c(x_i))$$

を最大化する μ_c と Σ_c を求めることで求まる。

$\mu_c^{(t+1)}$ と $\Sigma_c^{(t+1)}$ は以下のようになる。

$$\mu_c^{(t+1)} = \frac{\sum_{i=1}^{N} g_{ic}^{(t)} x_i}{\sum_{i=1}^{N} g_{ic}^{(t)}}$$

$$\sum_c^{(t+1)} = \frac{\sum_{i=1}^{N} g_{ic}^{(t)} \left(x_i - \mu_c^{(t+1)}\right)' \left(x_i - \mu_c^{(t+1)}\right)}{\sum_{i=1}^{N} g_{ic}^{(t)}}$$

3．3　モデル

Σ_cのすべての要素を求めるのは計算が困難である。また、パラメータの数が膨大のため推定されるモデル自体が複雑になり、学習の観点からも好ましくない。よって、簡略化されたΣ_cのモデルが提案されている。以下に紹介していく。

最も単純なモデルは、すべてのΣ_cを単位行列とするモデル（EII）である。

$$\sum_c = \sigma^2 \mathrm{I} = \begin{pmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \sigma^2 \end{pmatrix}$$

この場合、$f_c(x_i)$は以下の形になっている。

$$f_c(x_i) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left\{ -\frac{1}{2} \sum_{j=1}^{n} \frac{\left(x_j^i - \mu_{cj}\right)^2}{\sigma^2} \right\}$$

上記では見やすいように各ベクトルの要素を以下のようにしている。

$$x_i = \left(x_1^i, x_2^i, \cdots, x_n^i\right)^t$$

$$\mu_c = \left(\mu_{c1}, \mu_{c2}, \cdots, \mu_{cn}\right)^t$$

また、EMアルゴリズムによる更新式は以下のとおりである。

$$\sigma^{2(t+1)} = \frac{1}{nN} \sum_{i=1}^{N} \sum_{c=1}^{K} g_{ic}^{(t)} \sum_{j=1}^{n} \left(x_j^i - \mu_{cj}^{(t+1)}\right)^2$$

もう少し複雑なモデルとして、上記のσ^2を各クラスタごとに可変にしたモデル（VII）がある。

$$\sum_c = \sigma_c^2 \mathrm{I} = \begin{pmatrix} \sigma_c^2 & 0 & \cdots & 0 \\ 0 & \sigma_c^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \sigma_c^2 \end{pmatrix}$$

この場合、$f_c(x_i)$は以下の形になっている。

$$f_c(x_i) = \frac{1}{(2\pi\sigma_c^2)^{n/2}} \exp\left\{ -\frac{1}{2} \sum_{j=1}^{n} \frac{\left(x_j^i - \mu_{cj}\right)^2}{\sigma_c^2} \right\}$$

また、EMアルゴリズムによる更新式は以下のとおりである。

$$\sigma_c^{2(t+1)} = \frac{\sum_{i=1}^{N} g_{ic}^{(t)} \sum_{j=1}^{n} \left(x_j^i - \mu_{cj}^{(t+1)}\right)^2}{\sum_{i=1}^{N} g_{ic}^{(t)}}$$

すべてのΣ_cは共通として、すべての共分散を０、次元ごとに分散を可変にするモデル（EEI）がある。

$$\sum_c = \sum = \begin{pmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \sigma_n^2 \end{pmatrix}$$

この場合、$f_c(x_i)$は以下の形になっている。

$$f_c(x_i) = \frac{1}{(2\pi)^{n/2}\sqrt{\prod_{j=1}^{n}\sigma_j^2}} \exp\left\{-\frac{1}{2}\sum_{j=1}^{n}\frac{\left(x_j^i - \mu_{cj}\right)^2}{\sigma_j^2}\right\}$$

また、EMアルゴリズムによる更新式は以下のとおりである。

$$\sigma_j^{2(t+1)} = \frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{K} g_{ic}^{(t)}\left(x_j^i - \mu_{cj}^{(t+1)}\right)^2$$

上記のΣをクラスタごとに可変にするモデル（VEI）がある。

$$\sum_c = \begin{pmatrix} \sigma_{c1}^2 & 0 & \cdots & 0 \\ 0 & \sigma_{c2}^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \sigma_{cn}^2 \end{pmatrix}$$

この場合、$f_c(x_i)$は以下の形になっている。

$$f_c(x_i) = \frac{1}{(2\pi)^{n/2}\sqrt{\prod_{j=1}^{n}\sigma_{cj}^2}} \exp\left\{-\frac{1}{2}\sum_{j=1}^{n}\frac{\left(x_j^i - \mu_{cj}\right)^2}{\sigma_{cj}^2}\right\}$$

また、EMアルゴリズムによる更新式は以下のとおりである。

$$\sigma_{cj}^{2(t+1)} = \frac{\sum_{i=1}^{N} g_{ic}^{(t)}\left(x_j^i - \mu_{cj}^{(t+1)}\right)^2}{\sum_{i=1}^{N} g_{ic}^{(t)}}$$

第四章　次元縮約

高次元のデータをクラスタリングしようとすると、次元の呪いという問題が発生する。そのため、与えられたデータを高次元から低次元に変換する処理をする。それが次元縮約である。

n次元空間の点をk次元空間に変換する処理を射影という。n次元のデータがm個あるとき、このデータセットは行をデータに対応させると$n \times m$の行列Xで表現できる。データをk次元空間に射影する１つの方法が、$n \times k$の行列AをXに掛けることである。このとき、XAの行ベクトルがk次に縮約されたベクトルとなる。次元縮約を行うには上記の行列Aを求めることになる。

次元縮約の手法として、「特異値分解（SVD）」「pLSI」「NMF」がある。それぞれの手法を示す。

４．１　特異値分解（SVD）

次元縮約を行う標準的な手法が特異値分解である。特異値分解は$n \times m$の行列Xを３つの行列U、Σ、V'の積に分解する。

$$X = U \Sigma V'$$

ここで、Uは$m \times r$、V'は$r \times n$である。また、Σは$r \times r$の対角行列である。i行i列の対角要素をλ_iとおくと、

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r \geq 0$$

の関係がある。またrはXのランク（rank）である。

特異値分解は元のベクトルとのノルムの誤差ができるだけ小さくなるように次元を縮約する。以下に、結論を示す。

- Uの列ベクトルはXの列ベクトルの張る空間の正規直交基底である。
- V'の行ベクトルはXの行ベクトルの張る空間の正規直交基底である。
- λ_iはUのi番目の列ベクトル（あるいはV'のi番目の行ベクトル）の基底としての重要度を表している。

扱うデータは行ベクトルで表現されているので、行ベクトルの次元を縮約する。このため、特異値分解のV'を使う。適当な値以上のλ_iを選んで、k次元に縮約する場合、V'の最初のk行のベクトルをとってきて、$k \times n$の行列を作る。その転置行列をV_kとおく。V_kは$n \times k$である。m個のデータがn次元で表現され、それを$m \times n$のXとしているので、XV_kにより$m \times k$の行列が得られる。その行ベクトルがk次元に縮約されたデータを表している。

4．2　pLSI

pLSI はProbabilistic Latent Semantic Indexingの略で、確率・統計的な枠組みで処理を行う。処理する文書ベクトルには重み付けが必要なく、単純に頻度を要素とするベクトルでかまわない。pLSI はAspectモデルと呼ばれるモデルを利用している。Aspectモデルとは文書と単語を結びつける潜在的なクラスを想定したモデルであり、文書dと単語wの出現を潜在的なクラスzを用いて以下のようにモデル化します。

$$p(w|d)=\sum_{z}p(w|z)p(z|d)$$

ベイズの定理から、

$$p(z|d)=\frac{p(z)p(d|z)}{p(d)}$$

であり、$p(d,w)=p(d)p(w|d)$なので、

$$p(d,w)=p(d)\sum_{z}p(w|z)\frac{p(z)p(d|z)}{p(d)}=\sum_{z}p(z)p(w|z)p(d|z) \quad (4.1)$$

となる。求めるものは、$p(z)$、$p(w|z)$、$p(d|z)$である。

次に次元縮約する場合は、潜在的なクラスを　個設定する。

$z_1,z_1,\cdots,z_K$

データ（文書）dに対して、

$$\left(p(z_1,d),p(z_2,d),\cdots,p(z_K,d)\right)$$

が縮約されたベクトルとなる。

また、潜在的なクラスをそのままクラスタリングにおけるクラスタだと捉えれば、次元縮約した結果自体がクラスタリングを表す。つまり、データdのクラスタ番号は以下で得られる。

$$\arg\max_{k}p(d,z_k)=\arg\max_{k}p(z_k)p(d|z_k)$$

問題は$p(z)$、$p(w|z)$、$p(d|z)$を求めることだが、一般的な式を求める必要はない。与えられた文書集合が$D=\{d_1,d_2,\cdots,d_N\}$であり、Dで使われている単語の集

合が $W=\{w_1, w_2, \cdots, w_N\}$ である場合、各 $k \in \{1,2,\cdots,K\}$ に対する　、各 k と $m \in \{1,2,\cdots,M\}$ に対する $p(w_m \mid z_k)$、各 k と $n \in \{1,2,\cdots,N\}$ に対する $p(d_n \mid z_k)$ が求まればよい。つまり、全部で $K(1+M+N)$ 個の未知数（パラメータ）を求める。

これらのパラメータは最尤法で求めることができる。今、文書 d に含まれている単語 w の個数を $n(d,w)$ で表すと、対数尤度関数 L は以下のようになる。

$$L=\sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j)\log p(d_i, w_j)$$

式（４．１）を使って、以下が得られる。

$$L=\sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j)\log\left(\sum_{k=1}^{K} p(z_k)\,p(w_j \mid z_k)\,p(d_i \mid z_k)\right)$$

L を最大化するパラメータを求めればよいが、そのためにはEMアルゴリズムを用いる。ただし、尤度の関数に隠れ変数 z がすでに埋め込まれた形になっているので、Q 関数は直接求まる。

まず、E-stepでは z 以外のパラメータを固定したときの z の分布を求めるが、これは $p(z \mid d,w)$ である。Aspectモデルでは、

$$p(d,w,z)=p(z)\,p(w \mid z)\,p(d \mid z)$$

が仮定されているので、結局 $p(z_k \mid d,w)$ は以下のようになる。

$$p(z_k \mid d,w)=\frac{p(d,w,z_k)}{p(d,w)}=\frac{p(z_k)\,p(w \mid z_k)\,p(d \mid z_k)}{\sum_{k=1}^{K} p(z_k)\,p(w_j \mid z_k)\,p(d_i \mid z_k)}$$

簡単のために$p(z_k|d_i,w_j)=Q_{ijk}$とおいておく。E-stepではこのQ_{ijk}を求めることになる。更新式の形で書くと以下のとおりである。

$$Q_{ijk}^{(t+1)}=\frac{p(z_k)^{(t)}\,p(w_j|z_k)^{(t)}\,p(d_i|z_k)^{(t)}}{\sum_{k=1}^{K}p(z_k)^{(t)}\,p(w_j|z_k)^{(t)}\,p(d_i|z_k)^{(t)}}$$

次に、M-stepではzを固定した場合、つまりQ_{ijk}を固定した場合のその他のパラメータにを求める。

$$L=\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)\log\left(\sum_{k=1}^{K}p(z_k)\,p(w_j|z_k)\,p(d_i|z_k)\right)$$

$$=\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)\log\left(\sum_{k=1}^{K}Q_{ijk}\frac{p(z_k)\,p(w_j|z_k)\,p(d_i|z_k)}{Q_{ijk}}\right)$$

$$\le\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)\sum_{k=1}^{K}Q_{ijk}\log\frac{p(z_k)\,p(w_j|z_k)\,p(d_i|z_k)}{Q_{ijk}}$$

最後の式の変形はJensenの不等式から得られる。そして最後の不等式では、等号が成立している。そのため、さらに変形して以下が成立する。

$$L=\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)\sum_{k=1}^{K}Q_{ijk}\log\left(p(z_k)\,p(w_j|z_k)\,p(d_i|z_k)\right)$$

$$-\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)\sum_{k=1}^{K}Q_{ijk}\log Q_{ijk}$$

上式の第２項はM-Stepでは定数なので、Lの最大化には関係がない。第２項を省いたものを再びLとおく。

$$L=\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)\sum_{k=1}^{K}Q_{ijk}\log\left(p(z_k)\,p(w_j|z_k)\,p(d_i|z_k)\right)$$

ここから、Lの最大化は$\sum_{i=1}^{N}p(d_i|z_k)=1$、$\sum_{j=1}^{M}p(w_j|z_k)=1$、

$\sum_{k=1}^{K} p(z_k)=1$の関係があるので、ラグランジェの未定乗数法を利用して解くことができる。

$$L+\sum_{k=1}^{K}\alpha_k\left(1-\sum_{i=1}^{N}p(d_i|z_k)\right)+\sum_{k=1}^{K}\beta_k\left(1-\sum_{j=1}^{M}p(w_j|z_k)\right)+\gamma\left(1-\sum_{k=1}^{K}p(z_k)\right)$$

$p(d_i|z_k)=u_{ik}$、$p(w_j|z_k)=v_{jk}$、$p(z_k)=w_k$とおいて、上記の式をu_{ik}、v_{jk}、w_kでそれぞれ偏微分して極値問題を解く。ここでQ_{ijk}は正確に書くと、$Q_{ijk}^{(t)}$であり、M-stepの時点で定数になっていることに注意する。

まず、u_{ik}の極値問題を解く。

$$\frac{\sum_{j=1}^{M}n(d_i,w_j)Q_{ijk}}{u_{ik}}-\alpha_k=0$$

よって、

$$u_{ik}=\frac{\sum_{j=1}^{M}n(d_i,w_j)Q_{ijk}}{\alpha_k}$$

両辺をiに関して和をとると、

$$\sum_{i=1}^{N}u_{ik}=\frac{\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)Q_{ijk}}{\alpha_k}$$

左辺は1なので、

$$\alpha_k=\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)Q_{ijk}$$

以上より、

$$u_{ik}=p(d_i|z_k)=\frac{\sum_{j=1}^{M}n(d_i,w_j)Q_{ijk}^{(t)}}{\sum_{i=1}^{N}\sum_{j=1}^{M}n(d_i,w_j)Q_{ijk}^{(t)}}$$

更新式の形で書くと以下のようになる。

$$p\left(w_j \mid z_k\right)^{(t)} = \frac{\sum_{i=1}^{N} n\left(d_i, w_j\right) Q_{ijk}^{(t)}}{\sum_{i=1}^{N}\sum_{j=1}^{M} n\left(d_i, w_j\right) Q_{ijk}^{(t)}}$$

左辺の繰り返し関数がtとなっているが、これは右辺の$Q_{ijk}^{(t)}$がすでに更新されている形である。

同様に、v_{jk}とw_kでそれぞれ偏微分して極値問題を解くことで、以下が得られる。

$$p\left(w_j \mid z_k\right)^{(t)} = \frac{\sum_{i=1}^{N} n\left(d_i, w_j\right) Q_{ijk}^{(t)}}{\sum_{i=1}^{N}\sum_{j=1}^{M} n\left(d_i, w_j\right) Q_{ijk}^{(t)}}$$

$$p\left(z_k\right)^{(t)} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M} n\left(d_i, w_j\right) Q_{ijk}^{(t)}}{\sum_{k=1}^{K}\sum_{i=1}^{N}\sum_{j=1}^{M} n\left(d_i, w_j\right) Q_{ijk}^{(t)}}$$

実際にEMアルゴリズムでパラメータを求める際には、初期値が必要である。つまり、$p\left(d_i \mid z_k\right)^{(0)}$、$p\left(w_j \mid z_k\right)^{(0)}$、$p\left(z_k\right)^{(0)}$を適当に設定しておかないといけない。$p\left(z_k\right)^{(0)} = 1/K$として、その他はランダムな値を与えることが多い。

4．3　NMF

NMF は Non-negative Matrix Factorization の略で、次元縮約を利用したクラスタリング手法である。想定されているデータはベクトル空間モデルで表現された文書データである。NMF では n 次元のデータを列ベクトルで表現する。データが m 個あるとき、データセットは $n \times m$ の行列 X で表せる。ベクトル空間モデルで表現された文書データの場合、この X は索引語文書行列と呼ばれる。

目的とするクラスタの数が K である場合、NMF では X を以下のような行列 U と V^t に分解する。

$$X = UV^t$$

ここで、U は $n \times k$、V は $m \times k$ であり、各要素は非負値となっている。

ここで、V の i 番目の行ベクトルが X の i 番目の列ベクトル（i 番目のデータ）を K 次元に縮約した結果である。

NMF では縮約後の各軸がクラスタとなる文書集合のトピックを表現していると考えられる。つまり、V の k 列目の要素の値が、k 番目のトピックとの関連度の大きさを表していると考えられる。そのため、i 番目の文書データのクラスタ番号は以下で得られる。

$$\arg\max_k v_{ik}$$

ここで、v_{ik} は行列 V の i 行 k 列の要素を表す。

NMF は縮約後の各軸をクラスタのトピックと捉え、その軸への射影した値が関連度の大きさを表していると考えられる。索引語文書行列 X が非負値であることから、縮約後の U や V を非負値にする。この条件のために、縮約後の各軸は LSI のように直交していない。しかし、縮約後の各軸にそのクラスタに共通して現れる単語が集まるようになり、より適切にクラスタリングが行えることが期待できる。

実際のクラスタリングでは、与えられた非負値行列 X とクラスタリング数 K から V と U を以下の繰り返しで求める。

$$V'_{ij} \leftarrow V_{ij} \frac{\left(X^t U\right)_{ij}}{\left(VU^t U\right)_{ij}} \quad (4.2)$$

$$U_{ij}' \leftarrow U_{ij} \frac{\left(X'V\right)_{ij}}{\left(UV'V\right)_{ij}} \quad (4.3)$$

ここで、U_{ij}とV_{ij}はそれぞれUとVのi行j列の要素を表す。また、$(Z)_{ij}$により行列Zのi行j列の要素を表す。上記の式により、現在のUとVから、U_{ij}'とV_{ij}'が得られる。つまり、新たなU'とV'が得られるので、それをUとVと見なして、上記の式を繰り返し適用する。

式（4．2）と式（4．3）の繰り返しで$\|X-UV'\|$の値（分解の誤差）は単調に減少してゆくことが示せるので、この繰り返しによりUとVを求めることができる。ただし、$\|X-UV'\|$の値は真の最小値に収束するとは限らない。上記の繰り返しで使われる初期値$U^{(0)}$と$V^{(0)}$によって収束先は異なる。つまり、上記の繰り返しで得られるUとVは局所解である。

式（4．2）と式（4．3）の繰り返しで$\|X-UV'\|$の値が単調に減少してゆくことを示す。

考え方は、現在のUとVをU'とV'に固定したときに、$\|X-UV'\|$を最小にするVを求め、次に現在のUとVをU'とV'に固定したときに、$\|X-U'V'\|$を最小にするVを求め、次に現在のUとVをU'とV'に固定したときに、$\|X-UV'\|$を最小にするUを求める。これを交互に繰り返すので、$\|X-UV'\|$の値が単調に減少してゆく。現在のUとVをU'とV'に固定したときに、$\|X-U'V'\|$を最小にするVは式（4．2）であることが示され、現在のUとVをU'とV'に固定したときに、$\|X-UV'\|$を最小にするUは、式（4．3）であることが示せる。

証明する前に１つ用語を定義しておく。

vをK次元の列ベクトルとし、$F(v)$はK次元から実数値へのある関数とする。このとき、以下の条件を満たす関数$G(v,v')$をFの補助関数と呼ぶ。

$$G(v,v') \geq F(v), \qquad G(v,v) \geq F(v)$$

まず、現在のUとVをU'とV'に固定したときに、$\|X-U'V'\|$を最小にするVは、式（４．２）から得られることを示す。読みやすくするためにUとU'の表記を交換して、固定している方をU、求める方をU'とおく。また、$H=V'$とおいて、Hのc番目の列ベクトルhを考える。

今、ある関数Fとその補助関数$G(v,v')$が存在するとする。ある$h^{(0)}$から開始して、次の手続きで$h^{(l)}$の列をえることを行う。ただし、$h^{(0)}$の各要素は０より大きい正の数である。

$$h^{(l+1)} = \arg\min_{h} G\left(h, h^{(l)}\right) \quad (4.4)$$

このとき、$F\left(h^{(l)}\right)$はlに関して減少関数になっているので、上記の手続きによって$F(h)$の局所最小値が得られる。この性質を利用する。

$F(h)$と$G(h,h')$を以下のように設定する。

$$F(h) = \frac{1}{2}\sum_{i=1}^{n}\left(x_i - \sum_{j=1}^{K} U_{ij}h_j\right)^2$$

$$G(h,h') = F(h') + (h-h')' \nabla F(h') + \frac{1}{2}(h-h')' K(h')(h-h') \quad (4.5)$$

ここで、x_iは行列Xのc番目の列ベクトルxのi番目の要素を表す。また、$K(h)$は$K\times K$の対角行列で、i行i列の要素は以下の値をとる。

$$\left(K(h)_{ii}\right)=\frac{\left(U^tUh\right)_i}{h_i}$$

$2F(h)$は$X-UH$のc番目の列ベクトルの各要素を自乗して足したものになっているので、$F(h)$を最小にするhが$\|X-UH\|$を最小にするHのc番目の列ベクトルとなっていることが分かる。一方、式（４．４）の更新により$F(h)$の局所最小値を与える$h^{(t+1)}$が得られる。そして、式（４．４）の更新が式（４．２）を表していることが示せる。証明はこの流れで行う。

まず最初に示すことは、上記で設定した$G(h,h')$は$F(h)$の補助関数である。

$G(h,h')$が$F(h)$の補助関数であることを示すためには、$G(h,h)=F(h)$は明らかなので、$G(h,h')\geq F(h)$を示せばよい。そこでまず、$F(h)$に対して以下の式が成立することを利用する。

$$F(h)=F(h')+(h-h')^t\nabla F(h')+\frac{1}{2}(h-h')^t\left(U^tU\right)(h-h') \quad (4.6)$$

式（４．５）と式（４．６）から$G(h,h')\geq F(h)$を示すには、

$$0\leq(h-h')^t\left(K(h')-U^tU\right)(h-h') \quad (4.7)$$

を示せばよいことがわかる。

実際に$K(h')-U^tU$は半正定値行列であることが示せるので、式（４．７）が成立し、結果として、$G(h,h')$が$F(h)$の補助関数であることが示せる。

次に、式（４．５）のhに$h^{(t)}$を入れて、$G\left(h,h^{(t)}\right)$を最小化する$h^{(t+1)}$を求める。これは以下のように求まる。

$$h^{(t+1)}=h^{(t)}-\left(K\left(h^{(t)}\right)\right)^{-1}\nabla F\left(h^{(t)}\right) \quad (4.8)$$

次に、式（４．８）を変形すると以下が得られる。

$$h_i^{(t+1)}=h_i^{(t)}\frac{\left(U^t x\right)_i}{\left(U^t U h^{(t)}\right)_i} \quad (4.9)$$

$\left(U^t x\right)_i=\left(U^t X\right)_{ic}$、$\left(U^t U h^{(t)}\right)_i=\left(U^t U H^{(t)}\right)_{ic}$なので、式（４．９）を各要素の形で書き直すと、

$$H_{ic}^{(t+1)}=H_{ic}^{(t)}\frac{\left(U^t X\right)_{ic}}{\left(U^t U H^{(t)}\right)_{ic}}$$

となる。今、$H=V^t$だったので、上の式を転値する。$\left(U^t X\right)_{ic}=\left(X^t U\right)_{ci}$、$\left(U^t U H^{(t)}\right)_{ic}=\left(V^{(t)} U^t U\right)_{ci}$なので、

$$V_{ij}^{(t+1)}=H_{ji}^{(t+1)}=V_{ij}^{(t)}\frac{\left(X^t U\right)_{ij}}{\left(V^{(t)} U^t U\right)_{ij}}$$

となる。これは式（４．２）を表している。

次に、現在の$U^{(t)}$と$V^{(t)}$を固定したときに、$\left\|X-U\left(V^{(t)}\right)^t\right\|$を最小にする$U$は、式（４．３）から得られることを示す。

これは先ほどの証明でUとHを逆転させて考えればそのまま示せる。

$$\left\|X-U\left(V^{(t)}\right)^t\right\|=\left\|\left(X-U\left(V^{(t)}\right)^t\right)^t\right\|=\left\|X^t-V^{(t)}U^t\right\|$$

の関係があるので、先ほどのXをX^t、先ほどの$U^{(t)}$を$V^{(t)}$、先ほどの$V^{(t)}$を$U^{(t)}$（つ

まりVをU）にすればよい。式（４．２）に代入して以下を得る。

$$U_{ij}^{(t+1)} = U_{ij}^{(t)} \frac{\left(XV^t\right)_{ij}}{\left(U^{(t)}V^tV\right)_{ij}}$$

これは式（４．３）を表している。

$X=UV^t$を満たす解は無数に存在することに注意する。もしもUとVが求める解になっている、つまり$X=UV^t$であれば、適当な$K\times K$の行列Dを持ってきて$H=UD$と$W=V\left(D^{-1}\right)^t$を作ると、

$$HW^t = UDD^{-1}V^t = UV^t = X$$

なので、HとWも求める解になっている。

実際に式（４．２）と式（４．３）の繰り返しでは、確かに$\left\|X-U^tV^t\right\|$の値は単調に減少するが、UとVの要素の値が、どんどん大きくなったり小さくなったりする。そのために通常、各繰り返しの後にUかVの大きさを正規化する。

$$U_{ij} \leftarrow \frac{U_{ij}}{\sqrt{\sum_i U_{ij}^2}}$$

また、NMF の更新式では初期値$U^{(0)}$と$V^{(0)}$が必要である。通常ランダムな正の値を与えるが、最終的に得られるUとVは初期値によって異なる。このため異なる初期値で複数UとVを求め、それぞれに対する分解の誤差の値を最も小さいものを選択することが行われる。

第５章　実験

５．１　データセットと評価

文書クラスタリングの対象となるデータセットを示す。５つのデータセットを用意し、クラスタリングを行った。データセットの詳細を表５．１に示す。

表５．１　データセット

データセット	文書数	単語数	正解クラスタの数
fbis	2463	2000	17
reo	1504	2886	13
re1	1657	3758	25
tr23	204	5832	6
tr45	690	8261	10

クラスタリング結果の評価にはPurityとEntropyを用いた。これらの値から次元縮約による混合分布モデルの手法が、従来のクラスタリング（k-means）より有効かどうかを判断する。

Purity

クラスタの純度を表す。正解クラスタとの一致の度合いであり、高いほうが精度が良いことを示す。

Entropy

クラスタの情報の曖昧さを表す。１つのクラスタに含まれる正解クラスタと一致しない文書のばらつきが多いと高くなる。Entropyは低いほうが精度が良いことを示す。

５．２　実験結果

実験結果を表と図にそれぞれ示した。モデルはEllを採用した。データによっては、k-meansと同じぐらいの評価があった。しかし、全体的に見るとk-meansより劣っている。pLSIについては極端に悪い結果が混じっている。

表５．２　クラスタリング結果の評価（Purity）

データセット	SVD	pLSI	NMF	kmeans
fbis	0.636216	0.2233049	0.5671945	0.6264718
re0	0.599734	0.6077128	0.6589096	0.6655585
re1	0.5938443	0.2697646	0.6041038	0.6240193
tr23	0.622549	0.6862745	0.622549	0.6911765
tr45	0.6724638	0.5942029	0.6057971	0.6942029

表５．３　クラスタリング結果の評価（Entropy）

データセット	SVD	pLSI	NMF	K-means
fbis	0.3897439	0.8482273	0.4587738	0.3793676
re0	0.4028107	0.4214224	0.3852992	0.3827241
re1	0.414069	0.7287704	0.4008318	0.3631824
tr23	0.5274589	0.463329	0.5237092	0.4639402
tr45	0.3832193	0.4641656	0.4651743	0.3748184

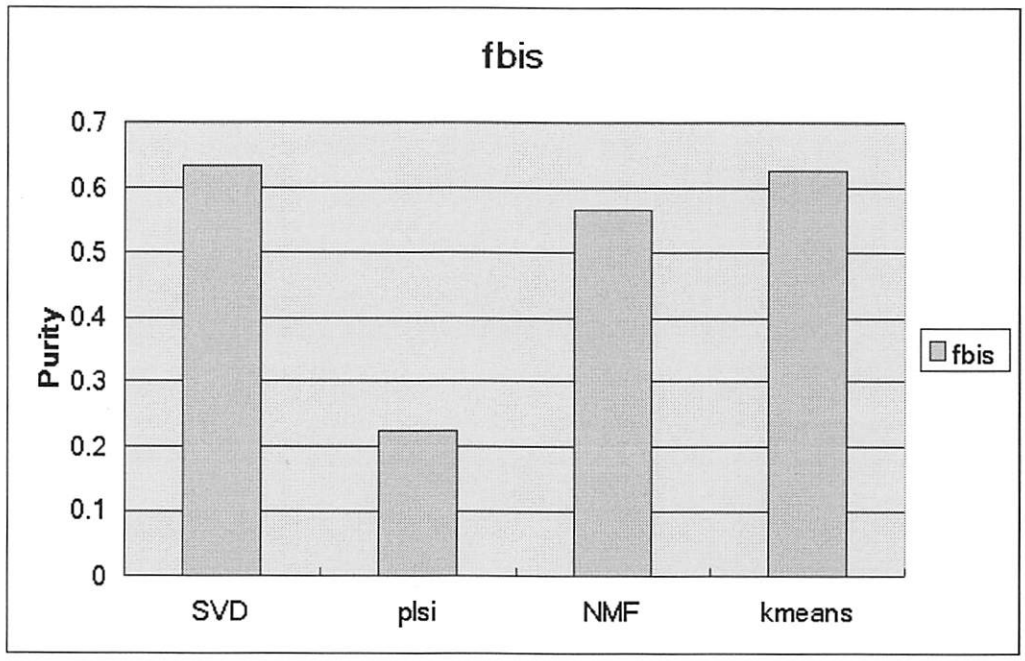

図５．１　クラスタリング結果の評価（Purity）

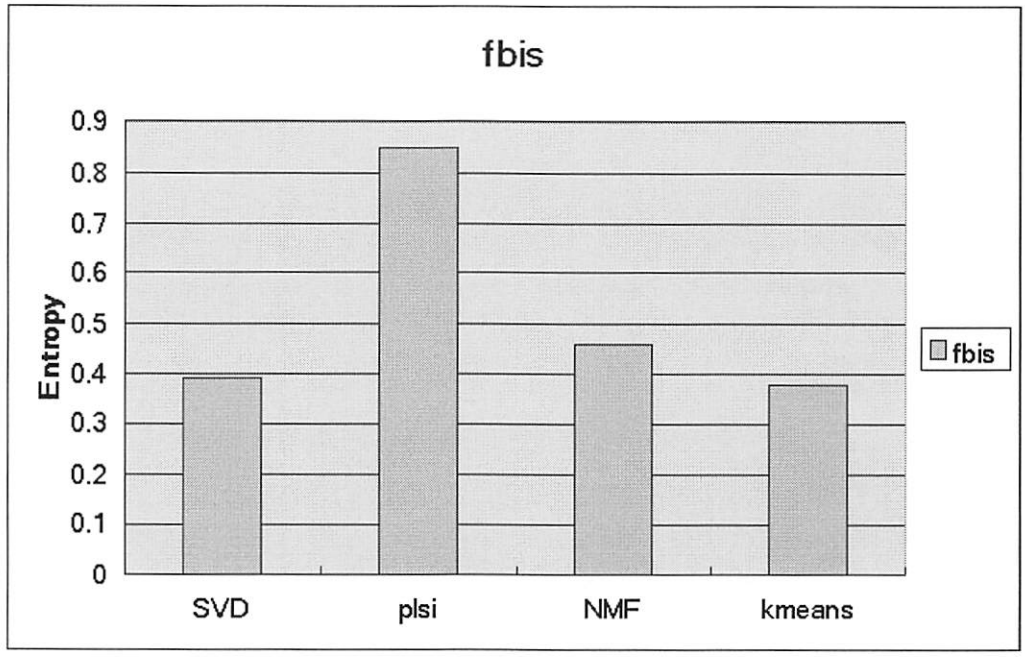

図５．２　クラスタリング結果の評価（Entropy）

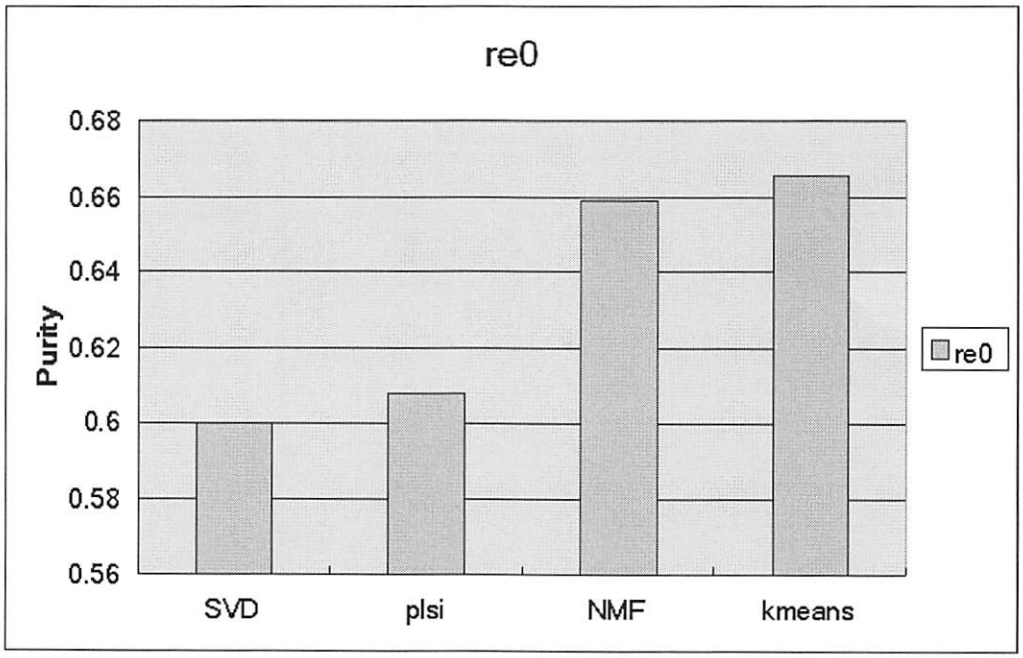

図５．３　クラスタリング結果の評価（Purity）

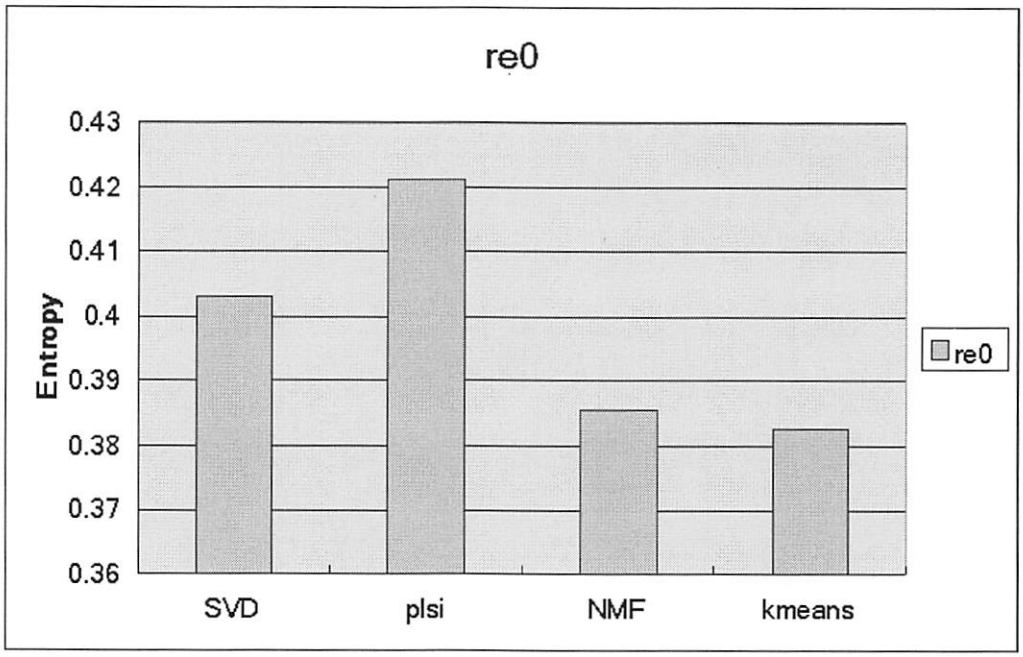

図５．４　クラスタリング結果の評価（Entropy）

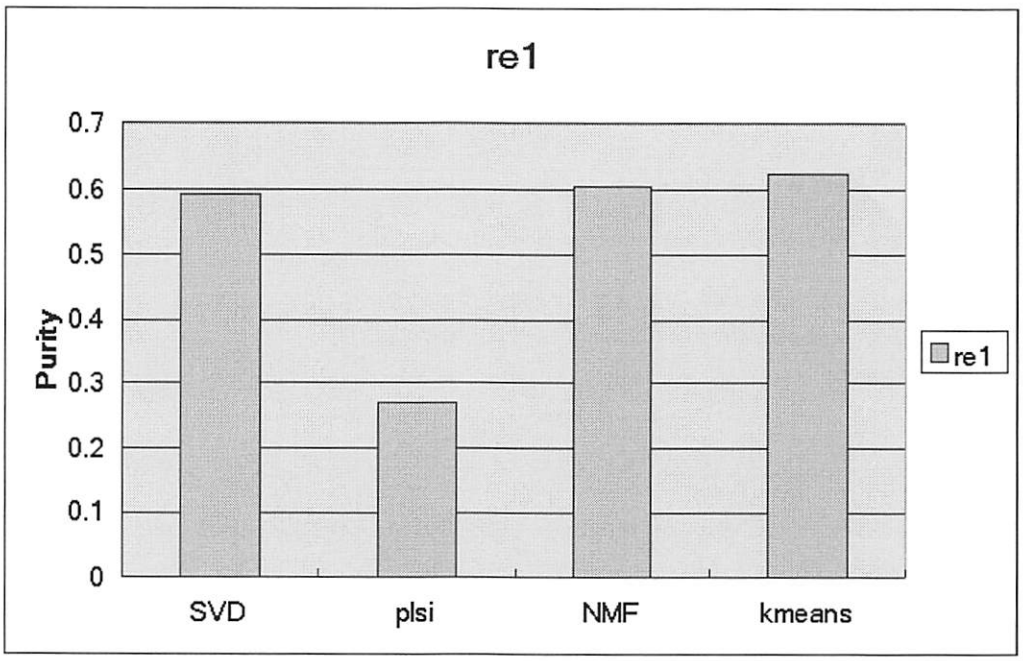

図５．５　クラスタリング結果の評価（Purity）

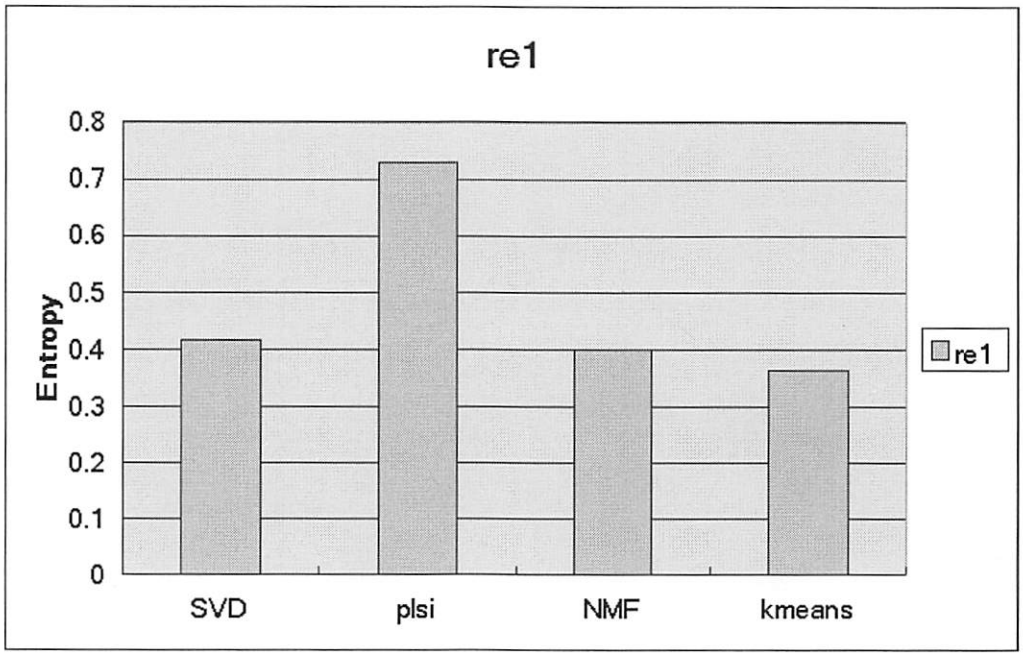

図５．６　クラスタリング結果の評価（Entropy）

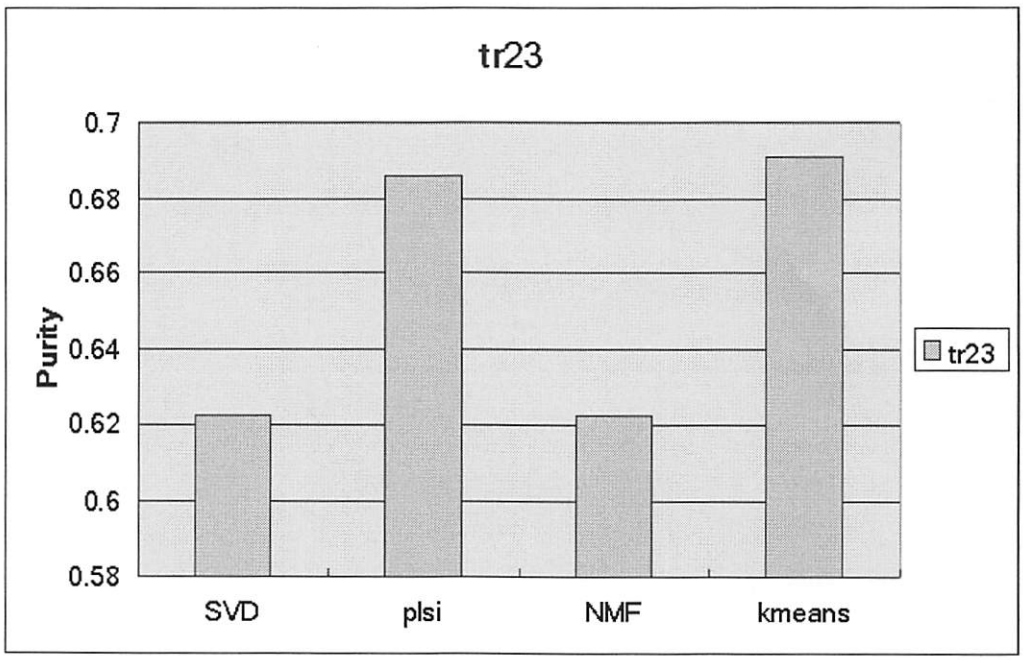

図５．７　クラスタリング結果の評価（Purity）

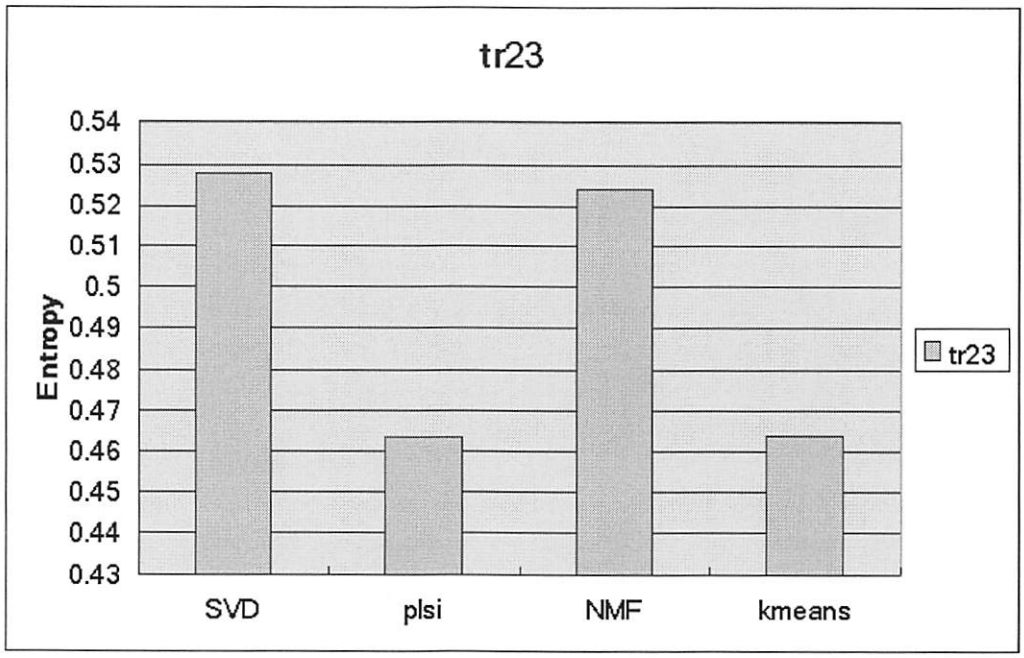

図５．８　クラスタリング結果の評価（Entropy）

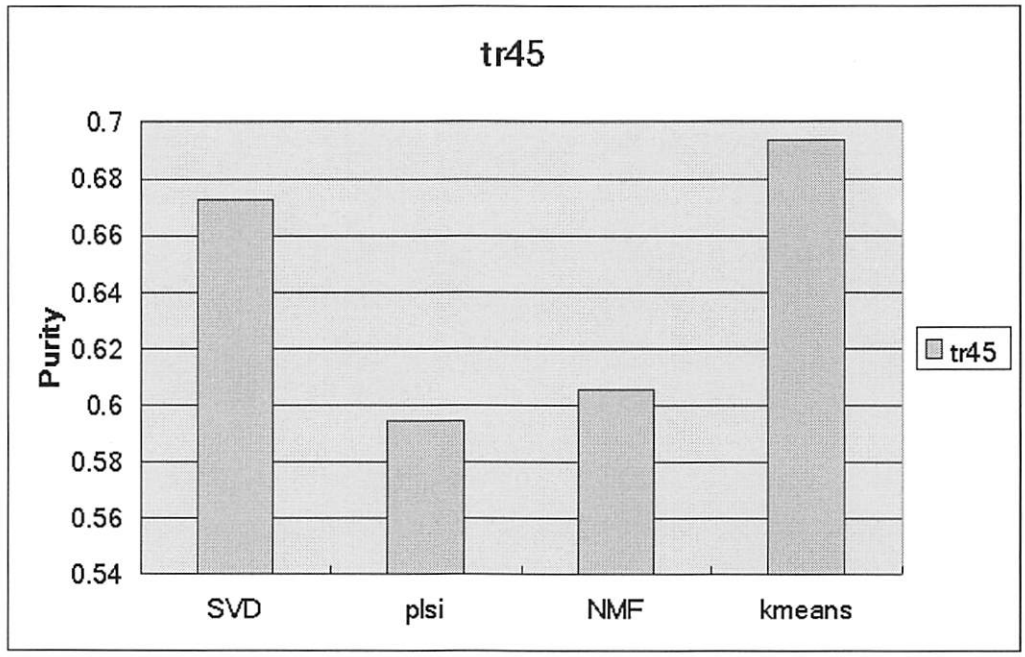

図５．９　クラスタリング結果の評価（Purity）

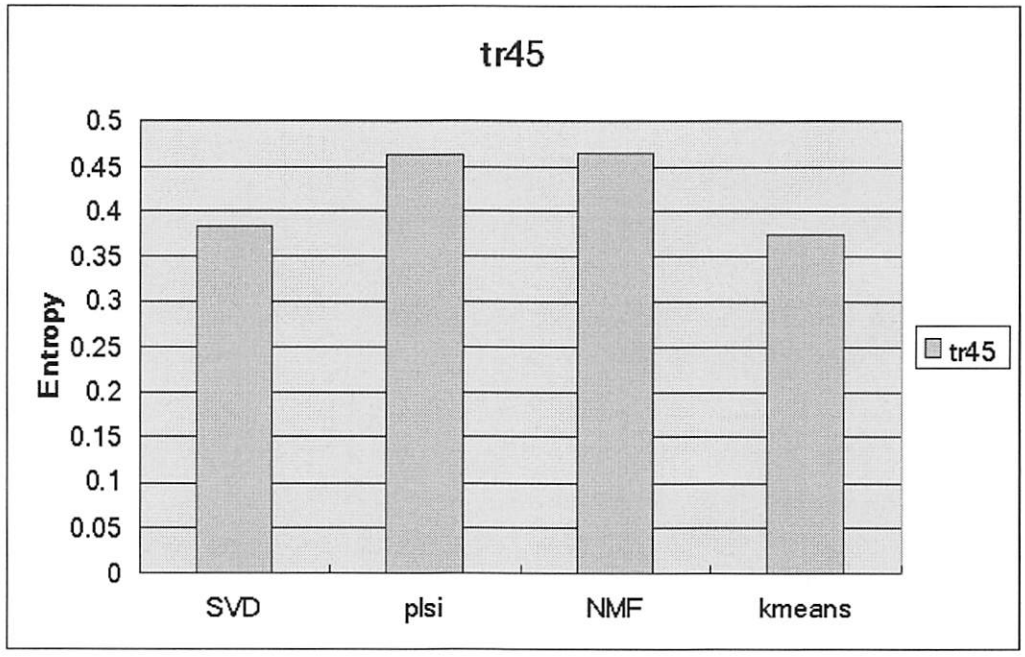

図５．１０　クラスタリング結果の評価（Entropy）

第6章　考察

実験結果は、次元縮約による混合分布モデルの手法がk-meansと比べて、同等程度の評価値が出たデータもあったが、精度的にはk-meansの方が優れていた。

この原因として、次元縮約後のデータが正規性をもっていないことが考えられる。なぜなら、混合分布モデルはデータが正規分布であることを前提にクラスタリングをおこなうため、次元縮約後のデータが正規性をもっていないとよい結果が得られないためである。そこで、分布が正規性をもっているかどうかの検証することにした。得られた分布が正規分布がどうかを検証する方法がある。次に正規性の検証法と検証結果を示していく。

６．１　正規性の検定法

正規分布への適合度の検定の手法を示す。（プログラムは付録にて）

検定手順

１．前提

- 帰無仮説 H_0 :「母分布は正規分布である」。
- 対立仮説 H_1 :「母分布は正規分布ではない」。
- 有意水準 α で両側検定を行う。

２．まず最初に、正規分布のパラメータを推定する。

（１）　n 個のケースが、k 個のカテゴリーに分類されているとする。

$$n=\sum_{i=1}^{k} f_i$$

（２）　各階級の中心点を X_i、観測度数を f_i とする。

（３）　母平均と母分散の推定値 Mean、Variance を推定する。

$$Mean=\bar{X}=\sum_{i=1}^{k} f_i X_i / n$$

$$Variance=V=\left\{ n\sum_{i=1}^{k} f_i X_i^2-\left(\sum_{i=1}^{k} f_i X_i\right)^2 \right\} \Big/ n^2$$

（４）　第 i 階級と第 i+1 階級の限点を X_i'、それに対する標準化得点を Z_i とする。

$$Z_i=\left(X_i'-\bar{X}\right)\Big/\sqrt{V}$$

（５）　各 Z_i から $Z<Z_i$ となる確率 P_i を求め、差をとることにより各階級の確率

$p_i = P_i - P_{i-1}\left(i = 2,3,\cdots,k-1\right)$ を求める。

$$p_i = \Pr\left\{Z < Z_1\right\}$$

$$p_k = 1 - \left(p_1 + p_2 + \cdots + p_{k-1}\right)$$

3．理論度数は、$E_i = np_i$ となる。

4．期待値が1以下のカテゴリーを併合する。併合後のカテゴリー数を m とする。

5．以下の式で検定統計量を計算する。

$$\chi_0^2 = \sum_{i=1}^{m} \frac{\left(O_i - E_i\right)^2}{E_i}$$

6．χ_0^2 は、自由度が $m-1-2$ の χ^2 分布に従う。

7．有意確率を $P = \Pr\left\{\chi^2 \geq \chi_0^2\right\}$ とする。

8．帰無仮説の採否を決める。

- $P > \alpha$ のとき、帰無仮説を採択する。「母分布は正規分布でないとはいえない」。
- $P \leq \alpha$ のとき、帰無仮説を棄却する。「母分布は正規分布ではない」。

６．２ 検定結果

SVD,pLSI,NMF それぞれの検定結果を表に示した。

表６．１　次元縮約 SVD による分布の検定結果

データ	有意確率 P ([x]はデータの列)			
fbis	[1]2.250475e-147	[2] 0.000	[3]2.977212e-41	[4]2.772027e-315
	[5] 0.000	[6]7.657419e-187	[7] 1.411204e-165	[8] 5.489254e-202
	[9] 1.105373e-08	[10]2.412766e-180		
re0	[1] 9.534075e-184	[2] 0.000	[3] 0.000	[4] 1.355328e-16
	[5] 3.565340e-07	[6] 1.424356e-07	[7] 5.741062e-45	[8] 2.621315e-62
	[9] 6.377220e-28	[10] 4.190773e-57		
re1	[1] 3.773534e-24	[2] 5.946974e-38	[3] 0.000	[4] 5.914661e-91
	[5] 1.535280e-217	[6] 1.212855e-255	[7] 6.918852e-19	[8] 6.536982e-278
	[9] 1.803004e-93	[10] 9.518688e-43		
tr23	[1] 0.000	[2] 3.458622e-17	[3] 3.359991e-13	[4] 3.744579e-10
	[5] 5.650950e-14	[6] 0.003212068	[7] 5.143180e-11	[8] 5.902305e-11
	[9] 3.731692e-13	[10] 0.004835074		
tr45	[1] 2.318456e-39	[2] 0.000	[3] 1.082392e-26	[4] 5.813754e-192
	[5] 1.256729e-16	[6] 9.402258e-97	[7] 2.015450e-18	[8] 1.096907e-04
	[9] 1.383019e-11	[10] 4.379833e-64		

表６．２　次元縮約 pLSI による分布の検定結果

データ	有意確率 P（[x]はデータの列）			
fbis	[1]2.236746e-05	[2] 1.275257e-04	[3]8.372775e-04	[4]1.618476e-05
	[5] 2.857057e-13	[6]3.058541e-11	[7] 9.229330e-18	[8] 2.953638e-07
	[9] 1.999078e-10	[10]6.056153e-05		
re0	[1] 0.000	[2] 0.000	[3] 0.000	[4] 0.000
	[5] 0.000	[6] 0.000	[7] 0.000	[8] 0.000
	[9] 0.000	[10] 0.000		
re1	[1] 0.006329919	[2] 3.992735e-10	[3] 1.791718e-07	[4] 0.01161413
	[5] 0.05462732	[6] 0.02640598	[7] 5.186702e-08	[8] 9.941623e-04
	[9] 0.04616838	[10] 0.2968845		
tr23	[1] 1.307817e-142	[2] 6.329982e-88	[3] 8.469061e-59	[4] 1.541125e-46
	[5] 9.727810e-50	[6] 8.761988e-80	[7] 6.385082e-51	[8] 5.708537e-74
	[9] 3.726065e-94	[10] 0.000		
tr45	[1] 2.548109e-253	[2] 5.962551e-152	[3] 0.000	[4] 1.375332e-239
	[5] 0.000	[6] 0.000	[7] 3.911994e-111	[8] 5.948206e-91
	[9] 0.000	[10] 0.000		

表６．３　次元縮約 NMF による分布の検定結果

データ	有意確率 P（[x]はデータの列）			
fbis	[1]0.00	[2] 0.00	[3]0.00	[4]0.00
	[5] 0.00	[6]0.00	[7] 0.00	[8] 0.00
	[9] 0.00	[10]0.00		
re0	[1] 0.000	[2] 0.000	[3] 0.000	[4] 0.000
	[5] 0.000	[6] 0.000	[7] 0.000	[8] 0.000
	[9] 0.000	[10] 0.000		
re1	[1] 0.000	[2] 0.000	[3] 0.000	[4] 0.000
	[5] 0.000	[6] 0.000	[7] 0.000	[8] 0.000
	[9] 0.000	[10] 0.000		
tr23	[1] 2.524044e-111	[2] 9.673247e-199	[3] 4.277390e-154	[4] 2.267934e-227
	[5] 9.337413e-278	[6] 1.874487e-110	[7] 7.621708e-116	[8] 3.719262e-305
	[9] 3.719262e-305	[10]5.401667e-190		
tr45	[1] 0.000	[2] 0.000	[3] 0.000	[4] 0.000
	[5] 0.000	[6] 0.000	[7] 0.000	[8] 0.000
	[9] 0.000	[10]1.692554e-250		

それぞれの検証結果をデータごとに平均した値を表にまとめた。有意水準が５％で検定を行うとすれば（α＝0.05)、表の値はすべて0.05以下なので、次元縮約したデータの分布は正規分布とはいえないことが確認された。

表６．４　データごとの有意確率Ｐの平均値

データセット	SVD	pLSI	NMF
fbis	1.105373e-8	0.0010642	0.00
re0	4.9896959e-8	0.000	0.000
re1	6.9188897e-20	0.04430246	0.000
tr23	0.0008047	1.5421616e-47	2.1268990e-111
tr45	0.000010969	5.948206e-91	1.692554e-251

第７章　おわりに

本実験では、次元縮約による混合分布モデルの文書クラスタリングが標準的な k-means の手法より有効であるか検証した。

５の文書データを用いて、次元縮約（SVD、PLSI、NMF）を行い、混合分布モデルによるクラスタリングを行った。その評価を k-means によるクラスタリングと比較したが、よい結果が得られなかった。その原因として、次元縮約後のデータが正規性をもっていなかったことが分かった。

謝辞

本研究の遂行及び論文の作成に多大なご助言及び指導を賜わった新納浩幸教官（茨城大学工学部情報工学科）に深い感謝の意を表します。また、本研究を進めるにあたり助言、協力を頂きました、佐々木稔教官（茨城大学工学部情報工学科）にも深く感謝します。

参考文献

[1]新納浩幸："Rで学ぶクラスタ解析",オーム社,(2007)
[2]"適合度の検定--正規分布への適合度の検定"
<http://aoki2.si.gunma-u.ac.jp/lecture/GoodnessOfFitness/normaldist.html>,
(2009/2/6アクセス)
[3]R. Kannan, H. Salmasian, and Santosh Vempala,
"The Spectral Method for General Mixture Models",
Conference on Learning Theory (COLT) (2005).

付録

すべてR言語のプログラムである。

```
#####################################################################
library(Matrix)
library(mclust)

#SVDによるクラスタリング手順
mysvd <- function(infile,n,ansfile){
        x <- as.matrix(readMM(f))
        xsvd <- svd(x)
        v <- t(xsvd$v)[c(1:10),]
        rx <- x %*% t(v)
        mc <- Mclust(rx,G=n,modelName="EII")
        myeval(mc$classification,ansfile)
}

#pLSIのクラスタリング評価
myplsi <- function(infile,n,ansfile){
         rx <- matrix(scan(infile),ncol=10,byrow=TRUE)

        mc <- Mclust(rx,G=n,modelName="EII")
        myeval(mc$classification,ansfile)

        }


#NMFによるクラスタリング手順
mynmf <- function(infile,n,ansfile){
        source("mydefs.r")
        source("seikisei.r")
        x <- as.matrix(readMM(infile))
        v <- nmf(x,10)

        mc <- Mclust(v,G=n,modelName="EII")
```

```
        myeval(mc$classification,ansfile)
        myeval(mc4$classification,ansfile)

        seikisei(v)
}

#####################################################################

#NMF 本体
nmf <- function (tX,clsn,maxlc = 30) {
    X <- t(tX)
    n <- nrow(X)
    m <- ncol(X)
    U <- matrix(runif(n*clsn) + 0.1,ncol=clsn)
    U <- normalU(U)
    V <- matrix(runif(m*clsn) + 0.1,ncol=clsn)
    loopc <- 0
    while(loopc < maxlc) {
        v1 <- tX %*% U
        v2 <- V %*% t(U) %*% U
        Vn <- V * v1 / v2
        u1 <- X %*% V
        u2 <- U %*% t(V) %*% V
        Un <- U * u1 / u2
        V <- Vn
        U <- normalU(Un)
        loopc <- loopc + 1
    }
    return(V)
}

normalU <- function(U) {
    NN <- diag(1/apply(U,2,function(x) sqrt(sum(x**2))))
    return(U %*% NN)
}
```

```
#####################################################################


#正規性の検定手順
seikisei <- function(x){
        rx <- matrix(scan(x),ncol=10,byrow=TRUE)
        b <- c(1:10)
        a <- c(1:10)
        kekka <- c(1:10)
        i <- 1
        while(i<=10){
                b[i] <- list(hist(rx[,i],nclass=20))
                a[i]  <-  list(normaldist(b[[i]]$counts,  b[[i]]$breaks[1],
b[[i]]$breaks[2]-b[[i]]$breaks[1], 1))
                kekka[i] <- list(print(a[[i]]$result2))
                i <- i + 1
        }
}

# 正規分布への適合度の検定 (本体)
normaldist <- function( x,                                      # 度数ベク
トル
                        b,                                      # 最初の階
級の下限値
                        w,                                      # 階級幅
                        a)                                      # 測定精度
{
        n <- sum(x)                                             # データ数
        x <- c(0, x, 0)                                          # 上下にそ
れぞれ 1 階級を追加する
        k <- length(x)                                          # 階級数
        mid <- seq(b-w/2, b+k*w-w, w)-a/2                       # 級中心

        xbar <- sum(mid*x)/n                                    # 平均値
        variance <- sum(x*(mid-xbar)^2)/n                       # 分散 (不偏
```

```
分散ではない)
        SD <- sqrt(variance)                                    # 標準偏差
        result <- c("n"=n, "Mean"=xbar, "Variance"=variance, "S.D."=SD)

        z <- ((mid+w/2)-xbar)/SD                                # 級限界の
標準化得点
        p <- pnorm(z)                                           # 累積確率
        p[k] <- 1                                               # 最後の累
積確率は 1
        p <- p-c(0, p[-k])                                      # 各階級の
確率
        expectation <- n*p                                      # 各階級の
期待値
        table <- data.frame(mid, x, z, p, expectation)          # 結果をデ
ータフレームにする
        rownames(table) <- paste("c-", 1:k, sep="")

        while (expectation[1] < 1) {                            # 期待値が
1 未満の階級を併合
                x[2] <- x[2]+x[1]
                expectation[2] <- expectation[2]+expectation[1]
                x <- x[-1]
                expectation <- expectation[-1]
                k <- k-1
        }
        while (expectation[k] < 1) {                            # 期待値が
1 未満の階級を併合
                x[k-1] <- x[k-1]+x[k]
                expectation[k-1] <- expectation[k-1]+expectation[k]
                x <- x[-k]
                expectation <- expectation[-k]
                k <- k-1
        }
        chisq <- sum((x-expectation)^2/expectation)             # カイ二乗
統計量
        k <- k-3                                                # 自由度
```

```
        p <- pchisq(chisq, k, lower.tail=FALSE)                # P 値
        result2 <- c("chi-sq"=chisq, "d.f."=k, "P value"=p)
        return(list(result=result, table=table, result2=result2))
}



#####################################################################

#評価関数
myentropy <- function (ct) {
    -sum((rowSums(ct) / sum(ct))
          * apply(ct,1,function(pv) {
                        p1 <- pv/sum(pv)
                        p2 <- p1[p1 != 0]
                        sum(p2 * log(p2))
                      })) /log(ncol(ct))
}


mypurity <- function (ct) {
    sum(apply(ct,1,max)) / sum(ct)
}

myeval <- function(myans, ansfile) {
     goldans <- scan(ansfile,what="character")
     ct <- table(myans,goldans)
     cat("Entropy: ",myentropy(ct),"¥n")
     cat("Purity : ",mypurity(ct),"¥n")
}

#####################################################################
```