# ランダムプロジェクションによるテキストストリームの検索

大内　浩仁† 　三浦　孝夫† 　塩谷　勇††

† 法政大学 工学研究科 電気工学専攻　〒 184–8584 東京都小金井市梶野町 3-7-2
†† 産能大学 経営情報学部　〒 259–1197 神奈川県伊勢原市上粕屋 1573
E-mail: †{i03r3208,miurat}@k.hosei.ac.jp, ††shioya@mi.sanno.ac.jp

**あらまし**　テキストストリームの検索においては，差分情報をどのように取り扱うかが検索処理の効率向上において問題となる．本論文では，ランダムプロジェクションを用い差分情報に対して動的に次元縮小を行うことで，検索性能および検索時間の双方で充分実用的な処理が行えることを述べる．

**キーワード**　情報検索，テキストストリーム, ランダムプロジェクション，次元縮小

# Retrieval for Text Stream by Random Projection

Hirohito OHUCHI†, Takao MIURA†, and Isamu SHIOYA††

† Dept.of Elect.& Elect. Engr., HOSEI University　3-7-2, KajinoCho, Koganei, Tokyo, 184–8584 Japan
†† Department of Management and Information Science, SANNO University　1573, Kamikasuya, Isehara city, Kanagawa 259–1197 Japan
E-mail: †{i03r3208,miurat}@k.hosei.ac.jp, ††shioya@mi.sanno.ac.jp

**Abstract**　In powerful yet efficient retrieval for text stream, one of the problems is how to maintain incremental information. In this investigation, we show that retrieval method using random projection is nearly useful choice for dynamic dimensionality reduction. We examine some experiments and show efficiency for computation and memory reducing dimension to incremental documents.

**Key words**　Information retrieval, Text stream, Random projection, Dimensionality reduction

## 1. 前　書　き

近年，情報検索の分野において，時系列データへの要求が高まっている．ニュース記事に代表される文書の時系列データ，いわゆるテキストストリームから検索を行い，必要な情報を取り出す際にはいくつかの問題がある．

テキストストリームでは，差分情報としてテキストデータが無限に送られてくるため，全ての情報を格納するためには，無限の記憶域が必要になってしまう．また，テキストストリームに対する検索は実時間応答でなければならない．テキストストリームの連続的に更新される情報に対して，連続的に質問検索を行う必要がある．

一般に，テキスト集合に対する検索は主要な語を取り出し，これを用いたベクトル空間モデルに基づいて処理される [7]．テキストデータは語いの数だけ次元が存在し，一般的に数万から数十万次元の高次元データとなる．高次元データをそのまま扱うと，計算機容量の確保および実時間応答が困難になる．計算機容量の効率化と，実時間応答を実現するために，テキストデータの次元を縮小して格納する必要がある．

テキストデータにおける次元縮小の方式として Latent Semantic Indexing（LSI）[4], [8], [9] が知られている．LSI 技術では，特異値分解（SVD）を用いて検索空間の次元を縮小する．これにより検索精度を維持したまま次元を大きく縮小することができるため，検索効率と検索精度を両立することができる．

しかし，LSI 技術にはデータの更新に対して特異値分解のための再計算が必要となる．このため，LSI 技術をテキストストリームの差分情報に対応させるのは容易ではない．フォルディング・イン [2] を用いれば再計算の必要はないが，本質的に元の文書集合に対してごく少数の文書の追加を仮定したものであり，更新に従って検索精度は低下する．

本研究では，ランダムプロジェクション（RP）[9] を用いることによって，計算機容量および検索効率において効率的なテキストストリームの検索方式を提案する．RP 技術による次元縮小は計算処理が少なく，データの更新時に再計算を行う必要がないため，差分情報に対して動的に次元縮小を行うことができる．これにより，検索質問に対する実時間応答が可能となる．

RP 技術と LSI 技術を比較した研究では，画像およびテキストデータの次元縮小において，RP 技術と従来の次元縮小方式を考察している [3]．テキストデータにおいては，RP 技術が遙かに少ない計算量で，LSI 技術と遜色のない検索精度を得るこ

とが述べられている．しかし，ここでは次元縮小によるデータの歪みを評価基準としており，時系列データおよび質問検索の評価は行っていない．

2 章ではテキストデータの次元縮小方式として RP 技術と LSI 技術について述べ，両者の比較を行う．3 章で RP 技術を用いたテキストストリームの検索について述べる．4 章に実験結果を示し，5 章で結びとする．

## 2. テキストデータの次元縮小

テキストデータの次元を縮小する方法として，前章で挙げた RP 技術と LSI 技術について述べる．以下では，単語数 $d$ ，文書数 $N$ の単語・文書行列 $X$ を考える．行列の大きさは $d$ 行 $N$ 列であり，それぞれの列ベクトルが 1 件の文書を表している．行列 $X$ の $i$ 行 $j$ 列の要素 $X_{ij}$ は，文書 $j$ における単語 $i$ の頻度である．

### 2.1 Latent Semantic Indexing

Latent Semantic Indexing（潜在的意味索引付け，LSI）では，特異値分解（SVD）によって次元縮小のための射影行列を求める．単語・文書行列 $X$ の SVD は，次の式で表される．

$$X_{d\times N} = U_{d\times r} S_{r\times r} V^T_{r\times N} \tag{1}$$

行列 $U$ ，$V$ は直交行列で，それぞれの列ベクトルを左特異ベクトル，右特異ベクトルと呼ぶ．行列 $S$ は対角行列であり，$S_{11} \geqq S_{22} \geqq \cdots$ という性質を持つ．これらの対角要素を特異値と呼ぶ．特異値分解のための計算量は $O(dN^2)$ であることが知られている [5].

次に，単語・文書行列を $k$ 次元 ($k << d$) に縮小する．LSI 技術における次元縮小は，次の計算で行う．

$$X^{SVD}_{k\times N} = U^T_k X \tag{2}$$

$U_k$ は大きさ $(d \times k)$ の行列で，行列 $U$ から最初の $k$ 個の左特異ベクトルを抜き出したものである．k 個の左特異ベクトルは，最も大きな k 個の特異値に対応している．

検索を行うために，検索質問はベクトル $\mathbf{q}_{d\times 1}$ で表現される．これを先と同様に低次元空間に射影する．

$$\mathbf{q}^{SVD}_{k\times 1} = U^T_k \mathbf{q}_{d\times 1} \tag{3}$$

次元縮小された質問ベクトル $\mathbf{q}_{k\times 1}$ と文書ベクトルの類似度を測定し，文書の類似度を降順にソートすることで，検索結果をランキングとして表示する．

類似度は，質問ベクトルと文書ベクトルの余弦（cos）で定義する．文書集合の中から $i$ 番目の文書を調べる場合，

$$\cos\theta_{ki} = \frac{(\mathbf{q}^{SVD}, \mathbf{X}^{SVD}_i)}{|\mathbf{q}^{SVD}||\mathbf{X}^{SVD}_i|}$$

の値によって，検索質問に対する文書の類似度を求める．$\mathbf{X}^{SVD}_i$ は，$X^{SVD}$ の $i$ 番目の列ベクトルを意味する．類似度は 1 から-1 の値を取り，1 に近いほど質問と適合している．

データの次元縮小に伴い，誤差が生じる．SVD について，フロベニウス・ノルムに基づく誤差を保証する定理が知られている [9]．$d \times N$ 行列 $X$ におけるフロベニウス・ノルムの 2 乗 $||X||^2_F$ は，次の式で定義される．[5]

$$||X||^2_F = \sum_{i=1}^{d}\sum_{j=1}^{N} |x_{ij}|^2 \tag{4}$$

$X$ を特異値分解して得た行列 $U$，$S$，$V$ から最初の $k$ 個の列ベクトルを抜き出し，行列 $U_k$，$S_k$，$V_k$ を作成する．これらの行列から $X$ をランク $k$ で近似した行列 $X_k$ を以下の計算で求める．

$$X_k = U_k S_k V^T_k \tag{5}$$

このとき，$X_k$ と $X$ の間に次の関係が成り立つ．

$$\min_{rank(Y)=k} ||X - Y||^2_F = ||X - X_k||^2_F \tag{6}$$

式（6）は，LSI 技術の次元縮小によって生じる誤差がフロベニウス・ノルムの意味において最小限に抑えられることを示す．

### 2.2 ランダムプロジェクション

ランダムプロジェクション（RP）は要素をランダムに決定した行列である．これにより高次元データを低次元の部分空間に射影することができる．以下では，大きさ $d \times N$ の単語・文書行列 $X$ を大きさ $k \times N (k << d)$ の単語・文書行列 $X^{RP}$ に射影する．このため要素をランダムに並べた大きさ $k \times d$ の RP 行列 $R$ を決定する．単語・文書行列 $X$ の RP 技術による次元縮小は，次の計算で行う．

$$X^{RP}_{k\times N} = R_{k\times d} X_{d\times N} \tag{7}$$

この処理の計算量は $O(dkN)$ [9] である．すなわち，次元数を縮小するほど計算時間は短縮される．

RP 行列 $R$ の要素を構成する際には，2 つの制限がある．1 つは $R$ のそれぞれの列ベクトルが，単位ベクトルの長さと等しいことである．もう 1 つは，$R$ が直交行列であることである．しかし，行列の直交化は大きな計算量を必要とする．

そこで，非常に単純な要素の分布で 2 つの制限を近似的に満たす手法 [1] が提案されている．RP 行列 $R$ の要素 $r_{ij}$ は，次のような分布をとるように並ぶ．

$$r_{ij} = \sqrt{3} \cdot \begin{cases} +1 & \text{確率 } 1/6 \\ 0 & \text{確率 } 2/3 \\ -1 & \text{確率 } 1/6 \end{cases} \tag{8}$$

この分布に従う行列を作成するための計算量は $O(kd)$ であり，更に $k << d$ であることから，実際の処理時間は非常に少ない．

質問検索は LSI 技術と同じく，質問ベクトルをに低次元空間に射影して行う．

$$\mathbf{q}^{RP}_{k\times 1} = R\mathbf{q}_{d\times 1} \tag{9}$$

文書ベクトルとの類似度を計算し，検索結果としてランキングを決定する．

RP 技術の次元縮小による誤差は，ベクトル間のユークリッド距離に対して定義される．$d \times N$ 行列 $X$ から任意の 2 つの列ベクトルを取り出し，$\mathbf{x}_1$ および $\mathbf{x}_2$ と置く．$\mathbf{x}_1$ と $\mathbf{x}_2$ の

$d$ 次元におけるユークリッド距離は，$|\mathbf{x}_1 - \mathbf{x}_2|$ で定義される．RP 行列 $R$ により $k$ 次元に縮小された空間における $\mathbf{x}_1$ と $\mathbf{x}_2$ のユークリッド距離は以下の式で再現することができる．[3]

$$\sqrt{d/k}|R\mathbf{x}_1 - R\mathbf{x}_2| \tag{10}$$

式（10）が成り立つためには，$R$ が直交行列である必要がある．$R^TR$ が 単位行列に近似するほど，$R$ は直交行列に近くなる．$R$ の直交性に対する誤差を表す $d \times d$ 行列 $\epsilon$ を次の式で定義する．

$$\epsilon = R^TR - I \tag{11}$$

このとき，$\epsilon$ の要素は，平均 0，分散 $1/k$ の正規分布をとる [6]．よって縮小次元数 $k$ を大きくするほど，ユークリッド距離における誤差は減少する．

### 2.3 LSI 技術と RP 技術

LSI 技術，RP 技術共に射影行列を用いて次元縮小を行っている．LSI 技術が SVD によって射影行列を求めているのに対し，RP 技術では乱数の発生により射影行列を作成することができる．SVD が大きい計算量を必要とする処理であるのに対し，RP 行列は少ない計算量で作成することができる．

テキストストリームにおいては，差分情報が連続的に発生する．LSI 技術では元の文書・単語行列を用いて射影行列を作成するため，更新を行うためには必ず何らかの再計算を行う必要がある．テキストストリームの検索を行うためには，射影行列の更新が必要となる．対して RP 技術では，RP 行列がデータに依存しないため，更新された文書に対して同じ行列で射影を行うことができる．必要な計算処理は，差分情報の低次元空間への射影のみである．

LSI 技術，RP 技術によって生じる誤差の保証は，両方ともノルムの値に基づいている．LSI 技術では行列のノルムを指標とし，RP 技術はベクトル間のユークリッド距離を指標としている．LSI 技術が単語と文書の関係をできるだけ維持するのに対し，RP 技術は行列におけるベクトル間の相対的な関係を維持する．RP 技術がデータに依存せず，差分情報への処理だけで更新が可能なのはこのためである．コサイン類似度を用いた検索においては，同じ次元で RP 技術が LSI 技術と同等かそれ以上の検索精度を得る．

これらの点から，テキストストリームの検索において LSI 技術を使用するのは現実的ではない．RP 技術による検索は，実時間応答，計算機容量の 2 つの側面において，前述の問題を解決することができる．ただし，RP 技術による検索は，RP 行列が近似的にしか射影行列の条件を満たさないため，乱数の分布によって検索精度にばらつきが生じる．しかし，その分散は縮小次元数に反比例することが知られている [3] ので，次元数を一定以上に設定すればよいことが期待できる．

LSI 技術と RP 技術の処理時間，検索精度，および RP 技術の検索精度の分散について 4 章で実験を行い，テキストストリームの検索における RP 技術の有用性を実証する．

## 3. テキストストリームの検索処理

RP 技術によってテキストストリームを検索する際の更新情報への対応，および本研究で過去のデータをどのように取り扱うかを述べる．

### 3.1 更新処理への対応

一般的に，更新処理はある一定の時間間隔を持って行われる．その際，1 度に複数の文書が更新対象となるが，1 件ずつ処理を行う．更新文書ベクトル $\mathbf{d}$ に対して，

$$\mathbf{d}^{RP}_{k\times 1} = R\mathbf{d}_{d\times 1} \tag{12}$$

を計算し次元を縮小した後で格納することで，計算機容量を効率化する．

### 3.2 過去データの重み付け

テキストストリームでは，時系列的に配置されたテキストデータが順次入力されていくことになる．そのため，更新されたばかりの新しい文書と過去の文書が混在することになる．一般的に，質問者にとっては過去の情報より新しい情報に価値があることが多い．その場合，過去のデータと新しいデータを同列に扱うことはできない．

本研究では，過去のデータに対して指数関数的に減少する重みを用いる．テキストデータの日時から最新の時刻との差 $t$ を求め，指数関数のパラメータとして用いる．重み係数 $w_a(t)$ は，以下の式によって表す．

$$w_a(t) = \exp(-t/a) \tag{13}$$

$a$ は重み係数の減少度を調節するパラメータである．$a$ が大きいほど重み係数の減少は緩やかになり，$a = \infty$ では減少は止まる．類似度に重み係数を掛けることで，新しい文書と過去の文書の差別化を図る．

## 4. 実　　験

ここでは，以下で使用するテキストストリームの詳細と，検索質問に対する答えの評価について述べる．次に，LSI 技術と RP 技術の比較，RP 技術を用いたテキストストリームの検索を実験し，それらの実験結果について考察する．

### 4.1 実 験 環 境

実験で使用する計算機の構成を表 1 に示す．メモリ上に動的

| | |
|---|---|
| OS | FreeBSD 4.6.2 |
| CPU | Pentium4 2.8GHz |
| メモリ | 1GB |

表 1　計算機の構成

確保した配列を用いて，単語・文書行列を計算機上に格納する．

RP 行列を生成するための乱数の生成には，Mersenne Twister (注1) を用いている．

テキストデータには，Reuter-21578 (注2) を使用する．Reuter-21578 は，文書分類のためのテストコレクションとして構成されており，21578 件の新聞記事が収録されている．Reuter-21578 の中には本文の無い記事が存在するため，カテゴリおよび本文

(注1)：http://www.math.keio.ac.jp/ matumoto/mt.html
(注2)：http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

が記述されている 19042 件の記事データを使用する．

Reuter-21578 のカテゴリ名および記事本文を索引語とし，不要語（stop word）の削除および単語のステミング [7] を行う．結果として，26870 語の索引語を得る．これらの索引語のうち，1 回しか出現しない索引語が 9610 語ある．文書の検索においては，低頻度の索引語は文書の網羅性に欠けるため，索引語として適切ではない．一定の基準に基づいて低頻度の語を削除する事で，適切な索引語の集合を得ることができる．

このために，Zipf の法則 [10] を用いる．Zipf の法則は，文書中に現れる単語の出現頻度と出現頻度の順位との関係について経験的に成り立つ法則である．本研究では，Zipf の法則を適用した結果，2662 語の索引語を得ている．

テキストストリームとして用いるために，テキストデータを時系列順にソートする．更新間隔は 6 時間で，1 回の更新で最大 422 件，平均 95.7 件の更新が行われる．文書の更新がない区間を除き，全体で 199 回の更新処理を行う．

### 4.2 評価方法

#### 4.2.1 Zipf の法則

Zipf の法則には，高頻度の単語で成り立つ Zipf の第 1 法則と，低頻度の単語で成り立つ Zipf の第 2 法則がある．Zipf の第 1 法則は，単語の頻度 $f$ と頻度の順位 $r$ との積が定数 $C$ になるという法則である．

$$f \times r = C \tag{14}$$

Zipf の第 2 法則は，出現頻度 $f$ の単語の数 $F_f$ と頻度 1 の単語の数 $F_1$ との間に次の関係が成り立つという法則である．

$$\frac{F_1}{F_f} = \frac{f(f+1)}{2} \tag{15}$$

低頻度の単語をどの程度削除するかの基準として，まず「中程度の頻度」を決める必要がある．式（14）は高頻度の語に対して成り立ち，式（15）は低頻度の語に対して成り立つことから，両方の式が同時に成り立つような単語の頻度 $f = f_k$ を求めることで，索引語として望ましい中程度の頻度を得る．

式（14）が低頻度の語で成り立たないのは，同順位の語がない（$F_f$ が常に 1）という仮定をしているためである．2 つの式が成り立つような $f_k$ を求めるためには，式（15）において $F_{f_k} = 1$ を代入すればよい．これを $f_k$ について解くと，次の式を得る．

$$f_k = \frac{\sqrt{8F_1 + 1} - 1}{2} \tag{16}$$

ここで得られた出現頻度 $f_k$ が索引語の頻度順位において中間地点であることを仮定すれば，以下の手順で索引語数を決定できる．

（1） 出現頻度 $f_k$ を持つすべての語を索引語とする

（2） 第 1 順位から $f_k - 1$ 個の頻度を持つ語までのすべてを索引語とする．全部で $K$ 個の語があるとする

（3） $f_k + 1$ 以下の出現頻度の語のうち，上位 $K$ 個を索引語とする

本実験では，$F_1 = 9610$ を式（16）に代入し，$f_k = 138$ を得る．上の手順に従うと，$K = 1339$，適切な索引語数は 2660 語となる．Reuter-21578 から出現頻度 48 以上の単語を抽出すると，その単語数は 2662 語となる．最終的にこの 2662 語を索引語として用いる．

#### 4.2.2 11 点平均適合率

検索結果の評価として，11 点平均適合率を用いる．11 点平均適合率とは，0.0 から 0.1 刻みで 1.0 までの再現率における適合率の平均値である．

再現率は，検索漏れの少なさを示す尺度であり，

$$\text{再現率} = \frac{\text{検索された文書中の適合文書の数}}{\text{全文書中の適合文書の数}}$$

で表される．適合率は，検索ノイズの少なさを示す尺度であり，

$$\text{適合率} = \frac{\text{検索された文書中の適合文書の数}}{\text{検索された文書の数}}$$

で表される．再現率と適合率はトレード・オフの関係にある．理想的な情報検索システムでは再現率と適合率が共に 1 となる．しかし，実際には検索漏れを無くそうとすれば不適合文書が混じり，適合文書だけを取り出そうとすれば検索漏れが発生する．

適合文書として，次元縮小を行わない状態で質問検索を行い，その結果類似度 0.5 以上となった文書を選ぶ．これにより，次元縮小による検索精度への影響を調べることができる．

テキストストリームの評価にも同じく 11 点平均適合率を用いる．本実験では，更新単位である 6 時間分の文書が更新されるたびに検索質問を行い，11 点平均適合率を求める．データの更新による 11 点平均適合率の推移を調べ，その平均および最低適合率を求める．テキストストリームにおける適合文書は，次元縮小を行わない状態での検索質問で類似度 0.5 以上の文書である．

### 4.3 LSI 技術と RP 技術の比較

時系列を考慮しないデータを用いて，RP 技術と LSI 技術を使用した検索を行う．次元縮小の処理に必要な時間および次元縮小時の検索精度を実験により比較し，考察する．次節以降では，テキストストリームについての考察のみを行う．

#### 4.3.1 処理時間

LSI 技術による検索は，計算機容量の問題から，先頭の 10000 件のみを処理する．$2662 \times 10000$ の行列を特異値分解する際の処理時間は 21469 秒（6 時間弱）である．

RP 技術による検索では，19042 件の文書を 1 度に処理する．RP 行列の作成および次元縮小に要した時間を表 2 に示す．処理時間は次元数に比例している．

| 次元数 | 処理時間（秒） |
|---|---|
| 100 | 74 |
| 200 | 150 |
| 300 | 231 |
| 400 | 308 |
| 500 | 387 |

表 2　RP 技術の処理時間

処理時間の比較では，RP 技術が圧倒的に勝っている．処理時間に格差が生じた原因として，SVD が大きな計算量を必要とする処理であるとともに，元データに依存する処理であるこ

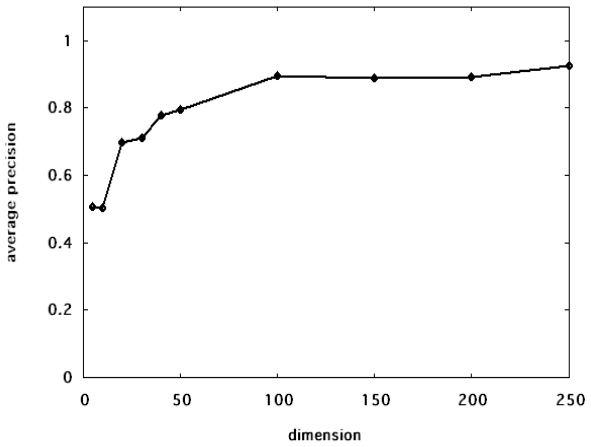


図 1　LSI 技術の検索精度

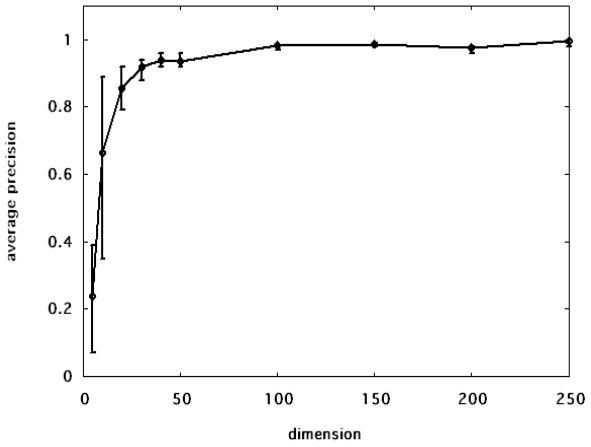


図 2　RP 技術の検索精度および検索精度の分散

とが挙げられる．縮小次元数を減らしても，SVD の所要時間は変化しない．他方，RP 行列は少ない計算量で作成が可能であるとともに，縮小次元数に応じて行列の大きさを減らすことができる．

#### 4.3.2　検 索 精 度

縮小後の次元数を 5 次元から 250 次元までの 10 段階に設定し，それぞれの次元数で LSI 技術と RP 技術による質問検索の検索精度を求める．適合文書数は，RP 技術が 635 件，LSI 技術が 407 件である．

RP 技術を用いた検索では，それぞれの次元で 3 回ずつ実験を行う．1 回毎に RP 行列を作り替えて次元縮小を行い，11 点平均適合率の平均値を最終的な評価とする．同時に最低値，最大値をプロットし，RP 技術による次元縮小で生じる分散を計る．

LSI 技術を用いた検索の結果を図 1 に，RP 技術を用いた検索の結果を図 2 に示す．

検索精度の比較では，平均値のみを考えれば 5 次元以外の全ての次元で RP 技術が上回っている．RP 技術の検索精度には

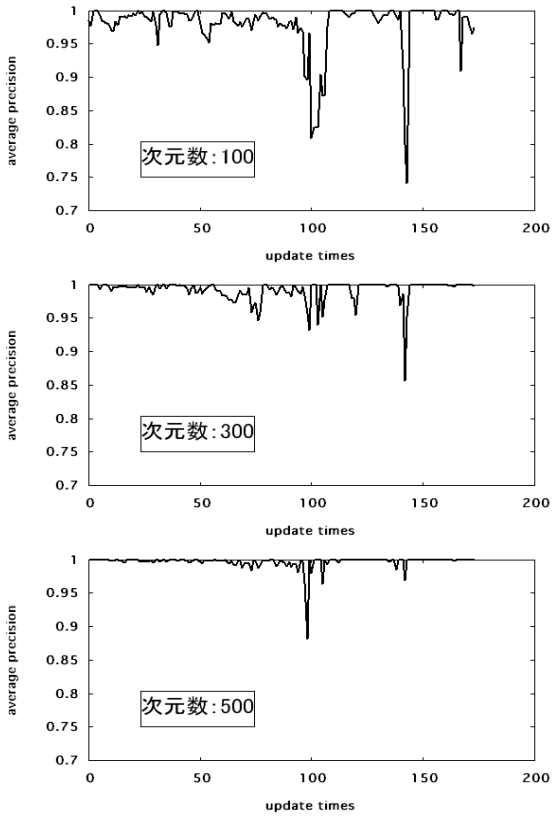


図 3　急激な重み付け（$a = 10$）の検索精度

分散が生じているため，単純に平均値だけで比較することはできない．分散は次元数が増加するほど少なく，100 次元以上では最低値と最大値の差が 1〜2 %に収束している．よって本実験においては，100 次元以上で RP 技術が安定して LSI 技術と同等以上の検索性能を発揮すると言える．

### 4.4　RP 技術によるテキストストリームの検索

データを時系列順にソートし，テキストストリームとして検索を行う．テキストストリームの検索では，式（13）に基づいて，過去のデータに対して重み付けを行う．$t$ の単位は日数とする．6 時間なら 0.25 である．

過去のデータに対する重み係数は次の 3 種類とする．

| | |
|---|---|
| 急激な重み付け | $(a = 10)$ |
| 緩やかな重み付け | $(a = 45)$ |
| 重み付けなし | $(a = \infty)$ |

急激な重み付けでは，重み係数は約 7 日間で 0.5，約 30 日間で 0.05 に減少する．緩やかな重み付けでは，重み係数は約 30 日間で 0.5，約 130 日間で 0.05 に減少する．重み付けなしの場合は，重み係数は常に 1 である．それぞれの重みで 100 次元，300 次元，500 次元の 3 つの次元数における検索を行う．

11 点平均適合率の推移をグラフで示す．急激な重み付けの検索結果を図 3 に，緩やかな重み付けの検索結果を図 4 に，重み付けなしの場合の検索結果を図 5 に示す．

総合的な指標として，更新毎に求めた 11 点平均適合率の平均値を求める．9 種類の実験における 11 点平均適合率の平均値を表 3 に示す．

最近の記事に類似文書が存在しない場合，更新時に適合文書

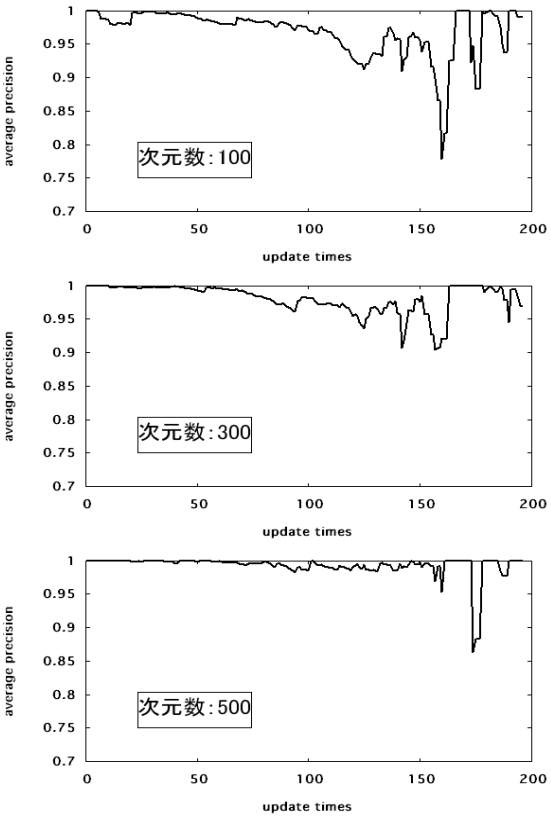


図 4　緩やかな重み付け（$a = 45$）の検索精度

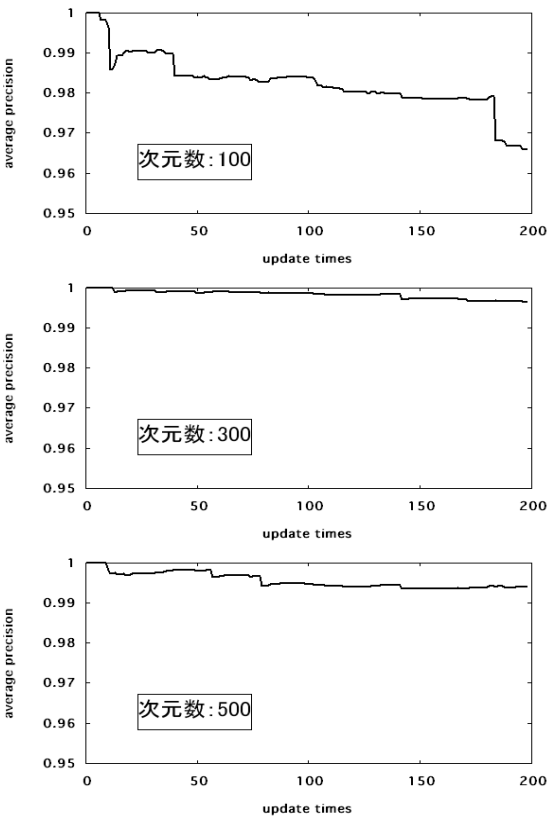


図 5　重み付けなし（$a = \infty$）の検索精度

| | 100 次元 | 300 次元 | 500 次元 |
|---|---|---|---|
| 急激な重み付け | 0.968 | 0.980 | 0.992 |
| 緩やかな重み付け | 0.979 | 0.992 | 0.997 |
| 重み付けなし | 0.982 | 0.998 | 0.995 |

表 3　11 点平均適合率の平均

が 0 件となることがある．その場合は無効質問として，グラフおよび 11 点平均適合率には反映させない．199 回の質問中に発生した無効質問の数を表 4 に示す.

| | 無効質問数 |
|---|---|
| 急激な重み付け | 25 |
| 緩やかな重み付け | 2 |
| 重み付けなし | 0 |

表 4　無 効 質 問

検索精度のばらつきを計るため，11 点平均適合率の最低値と最大値の差を求める．9 種類の実験全てで 11 点平均適合率の最大値が 1.0 であるため，最低値と最大値の差は検索誤差の最大値として得られる．表 5 に検索誤差の最大値を示す.

| | 100 次元 | 300 次元 | 500 次元 |
|---|---|---|---|
| 急激な重み付け | 0.259 | 0.143 | 0.119 |
| 緩やかな重み付け | 0.222 | 0.096 | 0.136 |
| 重み付けなし | 0.034 | 0.004 | 0.007 |

表 5　検索誤差の最大値

### 4.5 考　　察

11 点平均適合率の分布から，重み付けなしで類似度が 0.5 をわずかに上回るような低位の類似文書が適合率の低下を引き起こすと考えられる．更新によって低位の類似文書が増加すると適合率が減少し，重み付けにより低位の類似文書が類似文書から外れることで適合率が回復する．急激な重み付け（図 3）では，類似文書がすぐに入れ替わるため，検索精度の低下は局所的である．緩やかな重み付け（図 4）では急激な重み付けと比較して低位の類似文書が蓄積しやすく，検索精度の低下が継続して起こる箇所がある.

11 点平均適合率の平均値は，全ての場合で 90 %以上であり，更新に対して安定した検索を行っている．また，次元が多い方が概して平均値が高い．重み付けの種類で比較した場合は，重み付けなしの場合が最も高く，緩やかな重み付けが最も低い．しかし，重み付けがある場合には平均適合率の低下が局所的であるのに対して，重み付けなしの場合は低下が継続して起こっている.

無効質問の数は，重み付けなしの場合で最も少なく，急激な重み付けの場合で最も多い．重み付けを行う場合，質問に対する類似文書が，時間の経過によって適合文書から外れる．重み係数の減少が急激であるほど（$a$ が小さくなるほど）無効質問は発生しやすくなると考えられる.

## 5. 結　　論

本研究では，テキストストリームの検索において RP 技術を

適用した．RP 技術における誤差の保証から動的な検索処理が行えることを述べ，これにより検索効率と検索時間を充分実用的な範囲で両立したテキストストリームの検索が可能になることを示した．

本研究では限られた文書数のデータを対象としているが，実際のテキストストリームでは無限に情報が送られてくる．次元縮小によって必要となる記憶域は減少するが，無限数の文書を格納することはできない．

今後は，記憶域の利用についての議論を加え，実際のテキストストリームにおいて検索精度，検索時間，使用される記憶域の 3 点を議論する必要がある．

## 謝　　辞

本研究の一部は文部科学省科学研究費補助金 (課題番号 14580392) の支援による．

## 文　　献

[1] Achlioptas, D.: “Database-friendly random projections”, In *Proc. ACM Symp. on the Principles of Database Systems*, pp 274-281, 2001.

[2] Berry, M. W., Dumais, S. T. and O’Brien, G, W.: “Using linear algebra for inteligent information retrieval”, *SIAM Review*, Vol. 37, No. 4, pp. 573-595, 1995.

[3] Bingham, E. and Mannila, H.: “Random projection in dimentionality reduction: Applications to image and text data”, *Proc. 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001)*, pp 245-250, 2001.

[4] Deerwester, S. C., Dumais, S. T., Furnas, G. W., Landauer, T. K. and Harshman, R. A.: “Indexing by latent semantic analysis”, *journal of the American Society for Information Science*, Vol 41, No. 6, pp. 391-407, 1990.

[5] Golub, G. H. and Van Loan, C. F.: “Matrix Computations”, The Johns Hopkins University Press, 1989.

[6] Kaski, S.: “Dimensionality reduction by random mapping: Fast Similarity Computation for Clustering”, In *Proc. Int. Joint Conf. on Neural Networks (IJCNN)*, Vol 1, pp. 413-418, 1998.

[7] 北 研二, 津田 和彦, 獅子堀 正幹: “情報検索アルゴリズム”, 共立出版, 2002.

[8] 大内 浩仁, 三浦 孝夫, 塩谷 勇: “多義性を考慮した文書検索”, **データ工学ワークショップ（*DEWS*）**, 電子情報通信学会データ工学研究会, 2003.

[9] Papadimitriou, C. H., Raghavan, P., Tamaki, H. and Vempala, S.: “Latent semantic indexing: A probabilistic analysis”, In *Proc. 17th ACM Symp. on the Principles of Database Systems*, pp 159-168, 1998.

[10] Zipf, G, K.: “The human behavior and the principle of least effort”, Addison Wesley, 1949.