6B−7 量子化歪みの劣化がない高速ベクトル量子化

佐藤 龍哉 , 赤堀 一郎

日本電装株式会社

1 まえがき

音声や画像データの符号化の1つであるベクトル量子化は, 音声認識にも適用されるなど, 盛んに研究が進められている.

はじめに, ベクトル量子化を定式化する. 入力ベクトル $\boldsymbol{x}$ とコードベクトル $\boldsymbol{y}_i$ を k次元空間 R^k上の点とし, コードベクトルの集合 $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N\}$ をコードブックと呼ぶ. Nはコードブックのサイズである. ベクトル量子化は, 入力ベクトル $\boldsymbol{x}$ に対して, コードブックの中で最も距離 $d(\boldsymbol{x}, \boldsymbol{y}_i)$ が小さいコードベクトル$\boldsymbol{y}_i$を見つけることである. このコードベクトルのことを $\boldsymbol{x}$ の最近点と呼ぶことにする.

ベクトル量子化の課題の1つとして, コードブック探索時の計算量の削減がある. すべてのコードベクトルに対して距離計算を行なう全探索法(ＦＳＣ法)では, kN に比例した計算量が必要である. これまでに, 計算量の少ないアルゴリズムとして, 2分木を辿ってコードベクトルを求める2分木探索法(ＢＴＣ法)[1] が提案されている. しかし, ＢＴＣ法では, 必ずしも最近点を選択するとは限らないので, ＦＳＣ法と比較して量子化歪みが増加するという欠点がある.

ここでは, k次元ツリーでコードブックを表現することで, 少ない計算量で, 最近点が求められる(量子化歪みの劣化がない) ベクトル量子化方式を提案する.

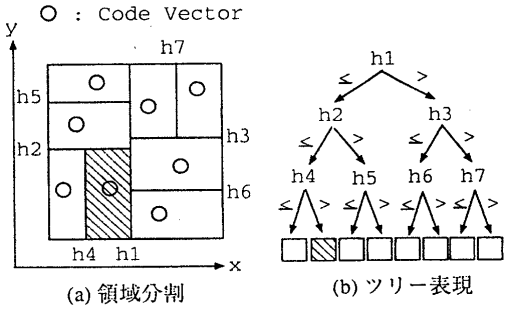
2 探索アルゴリズム

2.1 k次元ツリー

k次元ツリー [2] は, k次元空間を次に示す方法で次々と2分割していく2分木である. k次元ツリーのルートは, k次元空間全体に対応している. あるノードに対応している領域を Sとすると, 各ノード毎に定めた jと h を用いて, $S_l = \{\boldsymbol{x} \in S | x_j \leq h\}$ と $S_r = \{\boldsymbol{x} \in S | x_j > h\}$ とに分割し, それぞれを左ノードと右ノードに対応させる. 提案するアルゴリズムでは, jと h を最適 k次元ツリー [2] (jはその領域に含まれるコードベクトルの分散が最大になる軸, h はその軸成分の中央値) となるように選択し, 分割は, 領域がちょうど1個のコードベクトルを含むようになるまで続ける. k次元ツリーのリーフに対応する領域をバケットと呼ぶことにする.

Fast Vector Quantization Using a k-Dimensional Tree
Tatsuya SATO, Ichiro AKAHORI
Nippondenso Co., Ltd.

2次元の場合の例を図1に示す. (a) の斜線の領域は, (b) の斜線のリーフに対応している.



(a) 領域分割　(b) ツリー表現

図 1: k次元ツリー

2.2 アルゴリズムの構成

アルゴリズムは以下の3つの手順からなる.

手順1 質問点 $\boldsymbol{x}$ が存在するバケットを探索し, そのバケットに含まれるコードベクトルと $\boldsymbol{x}$ との距離 d を求める.

手順2 $\boldsymbol{x}$ を中心とする1辺 $2d$ の超立方体 Cと交わるバケットを列挙する.

手順3 列挙したバケットに含まれるコードベクトル $\{\boldsymbol{y}_i\}$ に対して, 木幡らのアルゴリズム [3] により, 最も近いコードベクトルを出力する.

質問点 $\boldsymbol{x}$ を含むバケットの探索は, k次元ツリーを使った2分探索によって, 容易に行なえる. しかし, そのバケットに含まれるコードベクトルは, 最近点であるとは限らない. 最近点は, $\boldsymbol{x}$ を中心とする半径 d の超球の内側に存在する.

Friedman らのアルゴリズム [2] では, 超球と交わるバケットを, バックトラックによってk次元ツリーを辿ることで探索する. 最近点は, 超球と交わるすべてのバケットに含まれるコードベクトルとの距離を計算することで求める. しかし, このアルゴリズムの問題点は, 超球とバケットの交わり判定の計算量が多いことである.

提案するアルゴリズムでは, 超球の代わりに, 超球に外接する超立方体を使い, これと各バケットとの交わりを判定する. 手順2では, コードベクトルとの距離計算は行わずに, 超立方体と交わるバケットの列挙のみを行う.

```
procedure  LIST(node) :
  if  node はリーフノードである then
      node に対応するバケットを候補リストに加える.
  else
      begin
          j ← node の j ;   h ← node の h ;
          if  x_j − h ≤ d  then  LIST(node の左ノード) ;
          if  h − x_j ≤ d  then  LIST(node の右ノード)
      end
```

図 2: 候補バケットを列挙する手続き

候補バケットの列挙は, 図2に示した手続き LIST を k次元ツリーのルートを引数として呼ぶことで行なう. ここで d は, 手順1で求めた値である. バケットを列挙する計算は, 次元数に依存しない. 超球の代わりに超立方体を使用したことで, 列挙するバケットの数は増加するが, 全体の計算量は削減できる.

最近点は, 列挙したバケットに含まれるコードベクトルに対して, 木幡らのアルゴリズムを適用することで求める. 木幡らのアルゴリズムは, $\boldsymbol{x}$ とコードベクトルの距離計算をコードブックの主成分軸上で $\boldsymbol{x}$ に近い順から行なうことで, 最近点を早い時期で見つけようとするものである. 最近点が発見できたかどうかの判定は, すべてのコードベクトルを調べなくても, 主成分軸上の距離がこれまでに調べた中で最も近いコードベクトルまでの距離以上になった時点で行なえる. 我々のアルゴリズムでは, 主成分軸の代わりに最も分散が大きい軸を使用する.

3 実験

１５次元ＬＰＣケプストラムの１０００個のベクトル量子化を行なった. コードブックの作成はＬＢＧアルゴリズム [4] を用いた.

実験では, コードブックのサイズを２５６として, 入力ベクトル1個あたりのベクトル間の距離計算 (DC) の回数, 各軸上の距離計算 (CDC) の回数 (CD 中の計算も含む), CPU 時間を測定した. 測定には SPARCstation2 を使用した. 実験結果を表1に示す.

比較したアルゴリズムは, 提案したアルゴリズム, Friedman らのアルゴリズム (FBF_BCK 法), 木幡らのアルゴリズム (SORT 法), FSC 法の4種類である.

表 1: アルゴリズムの比較

	DC	CDC	CPU
提案した方法	25.5 回	560.9 回	1.07mS
FBF_BCK 法	27.8 回	619.3 回	2.48mS
SORT 法	57.9 回	920.6 回	1.23mS
FSC 法	256.0 回	3840.0 回	3.33mS

FBF_BCK 法は, 超球とバケットの交わり判定のオーバーヘッドが多いので, 距離計算の回数が少なくても, CPU 時間がかかる. コードブックサイズ: ２５６, ベクトルの次元数:15 の条件で, 提案したアルゴリズムは, FSC 法に比べ CPU 時間で 1/3 程度になった.

4 まとめ

k次元ツリーのバケットと超立方体の交わり判定を行なうことで, 最近点の候補を列挙し, その中から木幡らのアルゴリズムで, 最近点を決定するベクトル量子化のアルゴリズムを提案した. 超立方体とバケットの交わり判定は, ベクトルの次元数に依存しない少ない計算量で行なえる.

今後は, ベクトルの次元数やコードブックのサイズや分布などの変動に対する影響の検討が必要である. また, 音声認識を行なう際のベクトル量子化に使用する予定である.

参考文献

[1] 鹿野清宏 : 入力音声のベクトル量子化による単語音声認識, 音響学会, 音声研究会資料, S82 60 (1982)

[2] Friedman, J. H., Bentley, J. L. and Finkel, R. A.: An Algorithm for Finding Best Matches in Logarithmic Expected Time, ACM Tran. Math. Software, Vol.3, No.3, pp.209-226 (1977)

[3] 木幡, 高木 : 主成分ベクトルを用いた高速ベクトル量子化, 電子情通学会論文誌 A, Vol.J71-A, No.2, pp.472 480 (1988)

[4] Linde, Y.,Buzo, A. and Gray, R. M.: An Algorithm for Vector Quantizer Design, IEEE Transaction on Communications, Vol.COM-28, No.1, pp.84-95 (1980)