1P2-lb-3in

2 進符号化を活用した高速かつ柔軟なクラスタリング

Fast Shape-Based Clustering Using Binary Encoding

杉山 麿人[*1*2] Mahito Sugiyama　山本 章博[*1] Akihiro Yamamoto

[*1]京都大学情報学研究科 Graduate School of Informatics, Kyoto University

[*2](独) 日本学術振興会特別研究員 DC2 JSPS Research Fellow

We present a new clustering algorithm for multivariate data, called BOOL (Binary cOding Oriented cLustering), that can detect arbitrarily shaped clusters and is noise tolerant. BOOL handles data using a two-step procedure: Every data point is first discretized and represented as a binary word; clusters are then constructed by agglomerating adjacent clusters using the binary representation. The latter step is performed in linear time with respect to the number of data points by sorting such representations. We experimentally show that BOOL is faster than K-means and about two orders of magnitude faster than two state-of-the-art algorithms that can detect non-convex clusters.

1. はじめに

クラスタリングは基本的なデータ解析手法であり，教師なし学習の 1 つとして機械学習や知識発見の分野でよく用いられる．特に K-means アルゴリズム [MacQueen 67] は，単純かつ高速なため，現在でも幅広く利用されている．しかし K-means は，超球状のクラスタのみしか対象としておらず，しかもノイズデータの存在を仮定していない．このため，任意形状のクラスタを発見可能で，かつノイズデータを検出可能なクラスタリング手法も数多く提案されてきた [Berkhin 06, Halkidi 01, Jain 99]．しかし，これらの手法のほとんどは，計算量が大きくデータサイズに対して 2 乗から 3 乗のオーダーであるため，大規模データへの適用は現実的でない．

そこで本稿では，K-means より速く，ノイズに対して頑健で，かつ任意形状を発見可能なクラスタリング手法 BOOL(Binary cOding Oriented cLustering) を提案し，その有効性を実験的に示す．近年，大規模データに適用可能な手法として SPARCL [Chaoji 09] や ABACUS [Chaoji 11] が提案されているが，本稿では，BOOL がこれらの手法よりも 100 倍以上速く，かつクラスタの質は同等であることを合成データ及び実データを用いた実験で示す．鍵となるアイデアは，データの **2 進符号化による離散化** (binary discretization) と，離散化されたデータのソートである．これによって，データサイズに対して線形時間でクラスタリングが実現できる．また，離散化の精度に応じてクラスタの階層が自然に導入されるため，BOOL は分枝型階層クラスタリングの一種となる．

単純な例を用いて，BOOL によるクラスタリングを概説する (図 1)．全てのデータが 2 次元の単位区間 $[0,1]\times[0,1]$ に入っていると仮定する．まず，データをレベル 1 (2 進符号で小数点以下 1 桁目) で離散化する．すなわち，各 x_i ($i=1$ または 2) は，$x_i \in [0, 0.5]$ ならば 0，$x_i \in (0.5, 1]$ ならば 1 となる．次に，離散化されたデータが同じ四角形の領域に入るか，または隣り合った四角形の領域に入っていれば同じクラスタとみなす (厳密な定義は定義 2 を参照)．したがって，レベル 1 では全てのデータが同じクラスタに属する．次に，データをレベル 2 で離散化する．各 x_i は，$x_i \in [0, 0.25]$ ならば 0，$x_i \in (0.25, 0.5]$ ならば 1，$x_i \in (0.5, 0.75]$ ならば 2，$x_i \in (0.75, 1]$ ならば 3 となる．ここでは，データ a からなるクラスタと，b,c,d,e からなるクラスタの 2 つが検出される．Naïve なアプローチでは，各データ間の距離を計算する必要があるため，このプロセスの計算量は $O(n^2)$ となる (n はデータ数)．しかし BOOL では，先に離散化されたデータをソートすることで，クラスタリングを線形オーダーで実現する．

id	元のデータ		レベル 1		レベル 2	
	x_1	x_2	x_1	x_2	x_1	x_2
a	0.14	0.31	0	0	0	1
b	0.66	0.71	1	1	2	2
c	0.72	0.86	1	1	2	3
d	0.48	0.89	0	1	1	3
e	0.74	0.48	1	0	2	1

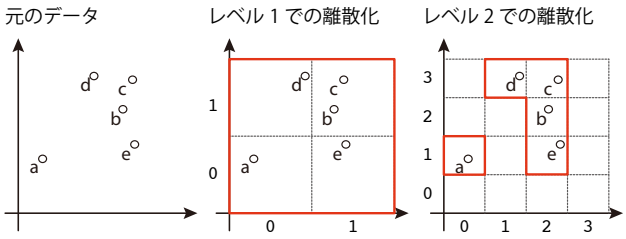


図1　BOOL によるクラスタリングの例.

本稿は以下のように構成されている．2 節で BOOL を導入し，3 節で実験により BOOL を評価し，4 節でまとめを述べる．

2. クラスタリング手法 BOOL

本稿では，クラスタリングの対象となるデータセットが関係データベース [Garcia-Molina 08] の形式で与えられると仮定する．各列 (カラム) を**属性** (attribute) と呼び，各属性の定義域を閉区間 $[0,1] \subset \mathbb{R}$ とする．実験においては，データ前処理 (min-max 正規化) によって各値を区間 $[0,1]$ 内の値へ変換する．また，属性は自然数 $1,\ldots,d$ である仮定する．したがって，各行 (タプル) は 1 つのデータに対応し，$\mathbb{R}^d$ の実ベクトル

連絡先: 杉山麿人，京都大学情報学研究科，606-8501 京都府京都市左京区吉田本町，075-753-5628，075-753-5628，mahito@iip.ist.i.kyoto-u.ac.jp

となる．各データ x に対して，i 番目の属性値を x_i と書く．また，属性の部分集合 $M \subseteq \{1,\dots,d\}$ に対して，$\pi_M(X)$ で X の M に含まれる属性のみからなるデータセットを表す．

クラスタリング（clustering）とは，データセット X を K 個の互いに素な部分集合（**クラスタ**）$C_1, C_2, \dots, C_K$ に分割することである．ここで，$C_i \neq \emptyset$，$C_i \cap C_j = \emptyset$ $(i \neq j)$，かつ $\bigcup_{i\in\{1,\dots,K\}} C_i$ が成り立つ[1]．この性質を満たすクラスタの集合 $\mathcal{C} = \{C_1,\dots,C_K\}$ を，X の**分割**（partition）と呼ぶ．クラスタ C の要素数を $\#C$ で表す．

BOOL 全体の擬似コードをアルゴリズム 1 に示す．準備として，まず 2 進符号化の方式に基づきデータの離散化を導入する．

定義 1 データ x の**レベル k での離散化**（discretization at level k）とは，以下のように x の各属性値 x_i を自然数 m に写す演算子 Δ^k である：$x_i = 0$ ならば $m = 0$，$x_i \neq 0$ ならば

$$m = \begin{cases} 0 & \text{if } 0 < x_i \leqslant 2^{-k}, \\ 1 & \text{if } 2^{-k+1} < x_i \leqslant 2^{-k+2}, \\ \dots & \\ 2^k - 1 & \text{if } 2^{-k+(k-1)} < x_i \leqslant 1. \end{cases}$$

データセット X 全体の離散化に対しても，同じ記号 Δ^k で表す．すなわち，X のデータ x は，$\Delta^k(X)$ では $\Delta^k(x)$ に離散化される．以下では，離散化レベル k を固定し，レベル k でのクラスタ構築を説明する．

次に，データ間の類似度を L_0 及び L_∞ 距離を用いて測る．データ x と y の間の距離は，それぞれ

$$d_0(x,y) = \sum_{i=1}^{d} \delta(x_i, y_i), \text{ ここで } \delta = \begin{cases} 1 & \text{if } x_i = y_i, \\ 0 & \text{if } x_i \neq y_i, \end{cases}$$
$$d_\infty(x,y) = \max_{i\in\{1,\dots,d\}} |x_i - y_i|$$

と定義される（δ はクロネッカーのデルタ）．

定義 2 データ x はデータ y から**レベル k で到達可能である**（reachable at level k）とは，データ $z_1, z_2, \dots, z_p$ $(p \geqslant 2)$ が存在して，$z_1 = x$，$z_p = y$，かつ任意の $i \in \{1,\dots,p-1\}$ に対して

$$d_0(\Delta^k(z_i), \Delta^k(z_{i+1})) \leqslant 1 \text{ かつ } d_\infty(\Delta^k(z_i), \Delta^k(z_{i+1})) \leqslant l$$

が成り立つことをいう．ここで，$l \in \mathbb{N}$ はあらかじめ与えられるパラメータであり，**距離パラメータ**と呼ぶ．

到達可能性は対称性を持つ．すなわち，データ y がデータ x からレベル k で到達可能であるとき，x は y からレベル k で到達可能である．直感的には，この条件は 2 つのデータに対して，値が異なっている属性の数が 1 以下で，かつその値の差が l 以下であることを意味する．また，$l = 1$ のときは

$$d_1(\Delta^k(z_i), \Delta^k(z_{i+1})) \leqslant 1$$

という条件と同値である（d_1 はマンハッタン距離）．

定義 3 データセット X の分割が**レベル k 分割**（level-k partition）であるとは，任意のデータ x と y に対して，y が x からレベル k で到達可能であるとき，そのときに限り同じクラスタに属することをいう．このとき，この分割を $\mathcal{C}^k$ と書く．

∗1 データの順序と重複を扱うため，データセットはタプルのリストとして扱う．一方，クラスタはデータの集合として扱い，順序と重複は考慮しない．

表1 データセット X と離散化されたデータセット $\Delta^1(X)$ 及び $\Delta^2(X)$.

X			$\Delta^1(X)$			$\Delta^2(X)$		
1	2	3	1	2	3	1	2	3
0.66	0.71	0.27	1	1	0	2	2	1
0.72	0.86	0.46	1	1	0	2	3	1
0.14	0.53	0.04	0	1	0	0	2	0

表2 データセット X と $Y = \pi_2(\Delta^2(X))$ によってソートされたデータセット $S_Y(X)$.

X			Y	$S_Y(X)$		
1	2	3	2	1	2	3
0.66	0.71	0.27	2	0.14	0.43	0.04
0.72	0.86	0.46	3	0.66	0.71	0.27
0.14	0.43	0.04	1	0.72	0.86	0.46

レベル k 分割は階層構造を持つ．すなわち，レベル k と $k+1$ 分割 $\mathcal{C}^k$ と $\mathcal{C}^{k+1}$ に対して，以下が成り立つ．任意のクラスタ $C \in \mathcal{C}^k$ に対して，$\bigcup \mathcal{D} = C$ を満たすクラスタの集合 $\mathcal{D} \subseteq \mathcal{C}^{k+1}$ が存在する．

例 1 表 1 のデータセット X を考え，x，y，z をそれぞれ 1 行目，2 行目，3 行目のデータとする．また，$l = 1$ とする．このとき，$d_0(\Delta^1(x), \Delta^1(y)) = 0$ かつ $d_\infty(\Delta^1(x), \Delta^1(y)) = 0$，$d_0(\Delta^1(x), \Delta^1(z)) = 1$ かつ $d_\infty(\Delta^1(x), \Delta^1(z)) = 1$，$d_0(\Delta^1(y), \Delta^1(z)) = 1$ かつ $d_\infty(\Delta^1(y), \Delta^1(z)) = 1$ となり，y と z はレベル 1 で x から到達可能なので，レベル 1 分割 $\mathcal{C}^1 = \{\{x, y, z\}\}$ となる．また，レベル 2 では x と y は z から到達可能ではないが y は x から到達可能なので，レベル 2 分割 $\mathcal{C}^2 = \{\{x, y\}, \{z\}\}$ となる．もし $l = 2$ であれば，レベル 2 分割は $\{\{x, y, z\}\}$ となる．

さらに，クラスタサイズの下限 N を導入することでノイズ除去も実現できる．この N をノイズパラメータと呼ぶ．レベル k 分割のクラスタ C に対して，$\#C < N$ ならば C に含まれるデータは全てノイズとする．例えば，例 1 のレベル 2 分割 $\mathcal{C}^2$ に対して，$N = 2$ のときクラスタ $\{z\}$ はノイズとなる．

レベル k 分割の構築に要する計算量は，naïve なアプローチでは，全てのデータのペアに対して L_0 及び L_∞ 距離を計算する必要があるため $O(n^2 d)$ となる（n はデータ数，d は属性数）．以下ではソートを用いることでこの計算量を減らす．まずはソートを定義する．

定義 4 データセット X と，X の属性 1 つだけからなるデータセット Y に対して，$S_Y(X)$ を Y の値で X をソートしたデータセットと定義する．Y の値が同じ場合は，X での順序をそのまま保存する．

この操作 S はクイックソートなどの標準的なアルゴリズムで実現できるが，本稿では離散化されたデータセット $\Delta^k(X)$ の値を用いたソートのみを扱う．各属性の定義域は 0 から 2^k であり，実際のクラスタリングにおいては k は高々10 程度である．このため，現在の標準的な計算機であれば，定義域の取りうる値全てをメインメモリにのせることができるので，バケットソートが利用できる．したがって，S によるソートの計算量は $O(n)$ となり，高速なソートが可能である．

アルゴリズム 1: クラスタリングアルゴリズム BOOL

Input: データセット X，クラスタ数（の下限） K，
ノイズパラメータ N，距離パラメータ l
Output: 分割 $\mathcal{C}$

function BOOL(X, K, N)
1: $k \leftarrow 1$ // k は離散化レベル
2: **repeat**
3: $\mathcal{C} \leftarrow$ MAKEHIERARCHY(X, k, N)
4: $k \leftarrow k+1$
5: **until** $\#\mathcal{C} \geqslant K$
6: output $\mathcal{C}$

function MAKEHIERARCHY(X, k, N)
1: $\mathcal{C} \leftarrow \{\{x\} \mid x \in X\}$
2: $h \leftarrow d$ // d は X の属性数
3: $X_{\mathrm{D}} \leftarrow \Delta^k(X)$ // X をレベル k で離散化する
4: $X \leftarrow S_{\pi_1(X_{\mathrm{D}})} \circ S_{\pi_2(X_{\mathrm{D}})} \circ \cdots \circ S_{\pi_d(X_{\mathrm{D}})}(X)$
5: **repeat**
6: $X \leftarrow S_{\pi_h(X_{\mathrm{D}})}(X)$
7: $\mathcal{C} \leftarrow$ AGGL(X, $\mathcal{C}$, h)
8: $h \leftarrow h-1$
9: **until** $h = 0$
10: $\mathcal{C} \leftarrow \{C \in \mathcal{C} \mid \#C \geqslant N\}$
11: output $\mathcal{C}$

function AGGL(X, $\mathcal{C}$, h)
1: **for each** X のデータ x
2: $y \leftarrow x$ の次のデータ
3: **if** $d_0(\Delta^k(x), \Delta^k(y)) \leqslant 1$ かつ
$d_\infty(\Delta^k(x), \Delta^k(y)) \leqslant l$ **then**
4: $\mathcal{C}$ から $C \ni x$ と $D \ni y$ を削除し $C \cup D$ を加える
5: **end if**
6: **end for**
7: output $\mathcal{C}$

例 2 表 2 のデータセット X を考え，$Y = \pi_2(\Delta^2(X))$ とする．このとき，$S_Y(X)$ は表 2 のようになる．

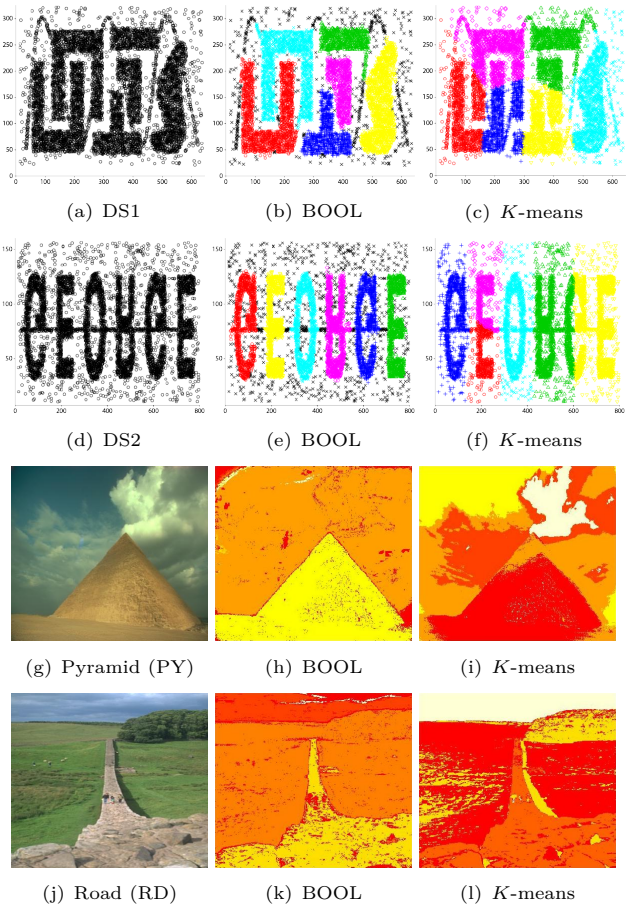
ソートを用いることで，各データと次のデータとの比較を $2d$ 回（関数 MAKEHIERARCHY の 4 行目で d 回，5 行目から 9 行目の d 回のループで 1 回ずつ）おこなうだけでクラスタリングが完了する．具体的には，データセット X とレベル k に対して，アルゴリズム 1 の関数 MAKEHIERARCHY(X, k, 0) の出力はレベル k 分割 $\mathcal{C}^k$ と一致することが示せる．ここで，関数 MAKEHIERARCHY(X, k, 0) の計算量は，関数 AGGL の 3 行目の条件判定が最良で $O(1)$，最悪で $O(d)$ となるため，最良で $O(nd)$，最悪で $O(nd^2)$ となる．したがって，BOOL 全体の計算量は最良で $O(ndk)$，最悪で $O(nd^2k)$ となる．ここで，$\#\mathcal{C}^{k-1} < K$ かつ $\#\mathcal{C}^k \geqslant K$ である．通常は $k \ll n$ が成り立ち，$O(nd)$ または $O(nd^2)$ となる．

3. 実験

合成データと実データを用いた実験によって，BOOL のスケーラビリティとクラスタリングの有効性を検証する．

3.1 実験手法

実験に用いた計算機は Mac OS X 10.6.5（2 × 2.26-GHz Quad-Core Intel Xeon CPUs, 12 GB）である．BOOL は C 言語で実装し，gcc 4.2.1. でコンパイルした．全ての実験は R 2.12.2 [R Development Core Team 11] 上でおこなった．



(a) DS1 (b) BOOL (c) K-means
(d) DS2 (e) BOOL (f) K-means
(g) Pyramid (PY) (h) BOOL (i) K-means
(j) Road (RD) (k) BOOL (l) K-means

図2 合成データセット DS1，DS2 と実データセット Pyramid (PY), Road (RD) に対する，BOOL 及び K-means によるクラスタリング結果．各実データセットは，3 つの属性（RGB）からなるデータセットに前処理で変換されている．また，実データセットに対するクラスタリング結果は等高線図で表されており，同じ色のピクセルは同じクラスタに属することを意味している．

合成データとして，CHAMELEON [Karypis 99] や SPARCL [Chaoji 09]，ABACUS [Chaoji 11] などのクラスタリング手法でベンチマークとして用いられてきたデータセットである，DS1 と DS2 を CLUTO[2]から入手し，評価に用いた（図 2(a), 2(d)）．BOOL によるクラスタリングの実行に際しては，どちらのデータセットにおいてもクラスタ数 $K = 6$ を指定し，パラメータ $(l, N) = (1, 100)$ とした．また，実データとして，Berkeley segmentation database and benchmark[3] [Martin 01] から 2 つの自然画像を入手した（図 2(g), 2(j)）．各画像は 481 × 321，すなわち 154,401 ピクセルからなり，前処理によって各ピクセルから RGB 値を得て，3 属性，154,401 データからなるデータセットとして扱った．これは，文献 [Chaoji 11] と全く同じ処理である．BOOL へのパラメータとして，Pyramid では $(l, N) = (5, 100)$，Road では $(l, N) = (15, 400)$ とした．また，これらのデータでは，クラスタ数が直感的には明らかでないので，関数 MAKEHIERARCHY を直接用いた．その

*2 http://glaros.dtc.umn.edu/gkhome/views/cluto/
*3 http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

表3　合成データセット及び実データセットにおける実行時間（秒）．表中の n と d はそれぞれデータ数と属性数を表す．BL, KM, DB, AB, SP はそれぞれ，提案手法 BOOL, K-means, DBSCAN, ABACUS (CHAMELEON), SPARCL (Random) を表す．ABACUS と SPARCL の結果は文献 [Chaoji 11] から引用した．クラスタリングの結果は図 2 に示す．

	n	d	BL	KM	DB	AB	SP
DS1	8000	2	**0.004**	0.008	9.96	1.7	1.8
DS2	8000	2	**0.004**	0.008	10.04	1.3	1.5
PY	154401	3	**0.187**	0.254	–	11.3	–
RD	154401	3	**0.180**	0.209	–	14.9	–

際，離散化レベルはそれぞれ 8 と 10 とした．

BOOL によるクラスタリングでは，全てのデータセットにおいて，各属性値が区間 $[0, 1]$ に入るように min-max 正規化 [Han 06] によって前処理をおこなった．この変換に要した時間も BOOL の実行時間に含めてある．

比較対象の手法として K-means[MacQueen 67] と DBSCAN[Ester 96] を採用した． K-meeans は標準的に用いられるクラスタリング手法であり，DBSCAN は任意形状のクラスタを抽出できる手法としてよく知られている．DBSCAN は R パッケージ `fpc` を用いて実行した．ただし，自然画像では DBSCAN は時間がかかりすぎてクラスタリングが完了できなかったため，結果は載せていない．さらに，BOOL の実行時間を最新のクラスタリング手法である ABACUS (CHAMELEON) [Chaoji 11] 及び SPARCL (Random) [Chaoji 09] と比較した．これらの手法のソースコードは公開されていないため，実装方法の違いによるバイアスを避けるため，本稿では文献 [Chaoji 11] に掲載されている値を引用し，比較をおこなった．用いたデータセットは文献 [Chaoji 11] で用いられたものとまったく同じであり，この比較は妥当である．これらのクラスタリング結果については，各文献を参照されたい．

3.2　結果と考察

実験結果を図 2 及び表 3 に示す．全ての合成データセットに対して，BOOL は最速であり，最新のクラスタリング手法 ABACUS 及び SPARCL より 100 ～ 1000 倍高速であった．さらに，BOOL は全てのクラスタを発見し，かつノイズの除去に成功した．ABACUS 及び SPARCL は任意形状のクラスタが検出可能であるが，ノイズの除去はできない．実データ（自然画像）に対しても同様に，実行時間は最速であり，かつ自然なクラスタの抽出に成功した．これらの結果は，空間的クラスタリングの適用対象として一般的な，属性数が小さいデータセットに対しては，BOOL が任意形状を抽出するクラスタリング手法として最速であることを示す．また，BOOL は階層クラスタリングとしても有効に機能することを示す．したがって，クラスタ数が不明な場合は，階層のレベルを直接入力して BOOL を利用することができる．

4.　おわりに

本稿では，新規のクラスタリング手法 BOOL を提案し，任意形状を抽出可能なクラスタリング手法としてこれまで提案されている手法のなかで最速であることを示した．鍵となるプロセスはデータの離散化とソートであり，これによってデータ数に対して線形時間でのクラスタリングを実現した．今後は，他の手法 [Ting 10] との比較や，生物学的データなどの大規模データへの適用が考えられる．また，異常値検出への適用や構造データへの適用，また半教師あり学習への応用も課題となる．BOOL は非常に単純なプロセスからなり，データの分布を仮定する必要がないため，さまざまな課題に対して有効に機能することが期待される．

謝辞

本研究の一部は，特別研究員奨励費（22 · 5714）の支援を受けている．

参考文献

[Berkhin 06] Berkhin, P.: A survey of clustering data mining techniques, *Grouping Multidimensional Data*, pp. 25–71 (2006)

[Chaoji 09] Chaoji, V., Hasan, M. A., Salem, S., and Zaki, M. J.: SPARCL: An effective and efficient algorithm for mining arbitrary shape-based clusters, *Knowledge and Information Systems*, Vol. 21, No. 2, pp. 201–229 (2009)

[Chaoji 11] Chaoji, V., Li, G., Yildirim, H., and Zaki, M. J.: ABACUS: Mining Arbitrary Shaped Clusters from Large Datasets based on Backbone Identification, in *Proceedings of 2011 SIAM International Conference on Data Mining*, pp. 295–306 (2011)

[Ester 96] Ester, M., Kriegel, H. P., Sander, J., and Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise, in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Vol. 96, pp. 226–231 (1996)

[Garcia-Molina 08] Garcia-Molina, H., Ullman, J. D., and Widom, J.: *Database systems: The complete book*, Prentice Hall Press (2008)

[Halkidi 01] Halkidi, M., Batistakis, Y., and Vazirgiannis, M.: On clustering validation techniques, *Journal of Intelligent Information Systems*, Vol. 17, No. 2, pp. 107–145 (2001)

[Han 06] Han, J. and Kamber, M.: *Data Mining*, Morgan Kaufmann, 2 edition (2006)

[Jain 99] Jain, A. K., Murty, M. N., and Flynn, P. J.: Data clustering: A review, *ACM Computing Surveys*, Vol. 31, No. 3, pp. 264–323 (1999)

[Karypis 99] Karypis, G., Eui-Hong, H., and Kumar, V.: CHAMELEON: Hierarchical clustering using dynamic modeling, *Computer*, Vol. 32, No. 8, pp. 68–75 (1999)

[MacQueen 67] MacQueen, J.: Some methods for classification and analysis of multivariate observations, in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1, pp. 281–297 (1967)

[Martin 01] Martin, D., Fowlkes, C., Tal, D., and Malik, J.: A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics, in *Proceedings of 8th International Conference on Computer Vision*, Vol. 2, pp. 416–423 (2001)

[R Development Core Team 11] R Development Core Team, : *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing (2011)

[Ting 10] Ting, K. M. and Wells, J. R.: Multi-Dimensional Mass Estimation and Mass-based Clustering, in *Proceedings of 10th IEEE International Conference on Data Mining*, pp. 511–520 (2010)