アルゴリズム 86－8
（2002．9．19）

地形図からの最適ピラミッドの構成アルゴリズム

全 眞嬉 (東北大学), 加藤直樹 (京都大学), 徳山 豪 (東北大学)

概要 地形図が与えられた時に、この地形図を近似する最適な「山」を計算する事を考えよう。具体的には、各地点の高度を表す2変数関数 $\rho(x,y)$ が与えられた時、これを、単峰関数、即ち極大点を一つだけ持つ関数 $f(x,y)$ で近似する。 本論文では、この問題を一般化し、最適化問題として定式化して考察する。この一般化した問題は、データマイニングや画像処理への応用を持っている。定式化された最適化問題は2変数の場合ですらNP完全問題となるが、出力関数に制限を与えることにより、様々な場合に効率の良いアルゴリズムを与える。

How to reform a terrain into a pyramid

Jinhee Chun[1], Naoki Katoh[2], Takeshi Tokuyama[3]

[1] GSIS, Tohoku University, Sendai, Japan. **(jinhee,tokuyama)@dais.is.tohoku.ac.jp**

[2] Graduate School of Engineering, Kyoto University, Kyoto, Japan. **naoki@archi.kyoto-u.ac.jp**

Abstract. Given nonnegative valued functions ρ and μ in d variables, we consider the optimal pyramid maximizing the total parametric gain of ρ against μ. The pyramid can be considered as the optimal unimodal approximation of ρ relative to μ, and can be applied to hierarchical data segmentation. We study properties of the optimal pyramid, and design efficient algorithms for several cases mainly for $d = 1$ and 2.

1 Introduction

Let ρ and μ be two nonnegative-valued functions defined on a cube $(0,n]^d$ in the d-dimensional Euclidean space $\mathcal{R}^d$. Naturally, we can consider them as distributions or measure functions on the cube. In particular, it is often convenient to regard them as measure functions to get intuition. Consider a family $\mathcal{F}$ of regions in the cube. For a region $R \in \mathcal{F}$, $g(R;\rho,\mu) = \rho(R) - \mu(R)$ is called the *gain* of ρ against μ in R where $\rho(R) = \int_{x\in R}\rho(x)dx$. More generally, introducing a nonnegative parameter t, $g_t(R,\rho,\mu) = \rho(R) - t\cdot\mu(R)$ is called the parametric gain of ρ against μ within the region R; this can be considered as the gain value in which we replace μ by $t\cdot\mu$.

The problem of finding the region $R \in \mathcal{F}$ maximizing $g(R;\rho,\mu)$ is a fundamental problem in data segmentation. Note that $\max_{R\in\mathcal{F}}|g(R;\rho,\mu)|$ is called the *discrepancy* between ρ and μ with respect to the family $\mathcal{F}$. In particular, the parametric gain is useful for solving segmentation problems in several applications including image processing and data mining; indeed, in those applications we seek for the segmentation maximizing a given concave objective function such as entropy or intercluster valiance, and the optimal segmentation also maximizes parametric gain for a suitable parameter value; this enables to design efficient algorithms applying standard methods of parametric optimization [1]. In data mining application for example, we select d numerical attributes to correspond a data record to a point in the cube ($d = 2$ in the literature [4, 7]). Given a large size of sample data set, we define μ to give the data distribution in the cube. Among the data set, we consider the subset of data that satisfy certain condition defined by a *target attribute*, and the subset gives another distribution function ρ. An *association rule* (named *d-dimensional rule* or *region tule*) is defined by using a region in R, so that μ and ρ determine *support* $\rho(R)$ and *confidence* $\rho(R)/\mu(R)$ of the rule. For obtaining a region yielding a rule with both large support and high confidence, we apply the above data segmentation.

In this paper, we would like to consider a parametric family of regions to give a good segmentation at each of parameter value with respect to the parametric gain. A *pyramid* $\mathcal{P}$ is a series of regions $\{P_t\}_{t\geq 0}$ in $\mathcal{F}$ satisfying that $P_t \subseteq P_{t'}$ if $t > t'$. Our aim is to compute the *optimal* pyramid $\mathcal{P}$ maximizing the total parametric gain $V(\mathcal{P}) = \int_{t=0}^{\infty} g_t(P_t; \rho, \mu)dt$.

The problem is more intuitive if μ is the unit function $\mu \equiv 1$. In this case, the optimal pyramid can be considered as a unimodal reformation of ρ minimizing loss of positional potential. This is a basic problem in computational geometry and geography (especially for $d = 2$). In general, the pyramid can be considered as a unimodal approximation of the measure ρ relative to μ. Figures 1 and 2 give examples of pyramids (where $\mu \equiv 1$) for $d = 1$ and $d = 2$, respectively. In Figure 2, function values are given by using density of pixels.

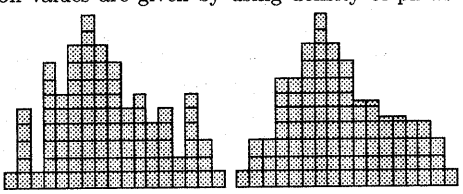

Figure 1: Input one variable function ρ and its optimal pyramid

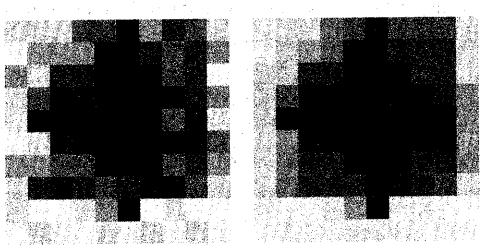

Figure 2: Optimal pyramid (right) for an input function ρ (left) in two variables

Construction of an optimal pyramid is a natural extension of region segmentation, and will be useful in several applications such as statistics, geomorphology, computer vision [2], and data mining; our main motivation is to develop a tool for constructing an optimized probabilistic decision tree in our project to extend the SONAR data mining system [3, 4, 6, 7].

In this paper, we study properties of optimal pyramids, and design efficient algorithms for computing them. In particular, we give an $O(n \log n)$ time algorithm for the one dimensional problem where $\mathcal{F}$ is the set of integral intervals in $(0, n]$, and also give efficient polynomial-time algorithms for several two dimensional cases.

2 Preliminaries

We consider functions on the cube $C = (0, n]^d$ that is decomposed into $N = n^d$ unit cubes (translated copies of $(0, 1]^d$) called *cells*. Here, for technical reason, we prefer the left-open and right-closed interval $(0, n]$ to the closed interval $[0, n]$.

The definition of a pyramid depends on the family $\mathcal{F}$ of regions, and we are interested in families $\mathcal{F}$ of regions in $C = (0, n]^d$ such that each region is a connected union of cells; in the other words, they are regions in the pixel grid G of size n^d. We also assume that $\mathcal{F}$ contains $\emptyset$.

A function f on C is called a *step function* if it is a constant function in each cell, that is, $f(x) = f(x')$ holds for any pair x and x' in the same cell. Since we only consider regions that are union of cells, we can assume without loss of generality that ρ and μ are step functions. Thus, ρ and μ are considered as d-dimensional real-valued arrays of size n^d.

We need to assume that $\mu(R) \neq 0$ in each region $R \in \mathcal{F}$ satisfying that $\rho(R) \neq 0$, since otherwise the objective function $V(\mathcal{P}) = \int_{t=0}^{\infty} g_t(P_t; \rho, \mu)dt$ is not bounded. For making the argument simple, we assume that $\mu(c) \neq 0$ for each cell c.

2.1 Properties of the optimal pyramid

We consider a pyramid on the cube $C = (0, n]^d$ with respect to a region family $\mathcal{F}$. For a pyramid $\mathcal{P}$, the *trajectory function* $f_{\mathcal{P}}$ is defined by $f_{\mathcal{P}}(x) = \sup\{t : x \in P_t\}$, which gives the boundary surface of the closure of the pyramid. We can observe that

the trajectory function is a unimodal function if each region in $\mathcal{F}$ is connected.

The horizontal section P_t of height t of a pyramid $\mathcal{P}$ can be represented by $\{x : f_{\mathcal{P}}(x) \geq t\}$. The *exposed part* of a section of height t is $\{x : f_{\mathcal{P}}(x) = t\}$. If the exposed part of a section is non-empty, we call it a *flat* of $\mathcal{P}$. The flat with the maximum height is called the *top flat*. By definition, a flat must be a set difference of two members of $\mathcal{F}$. A *balancing region* for t is $R \setminus R'$ of a pair $R \supset R'$ in $\mathcal{F}$ satisfying that $\int_{x \in R \setminus R}(\rho(x) - t \cdot \mu(x))dx = 0$. A flat of $\mathcal{P}$ is called a *balancing flat* if it is a balancing region for the height t of the flat.

Lemma 2.1 *For the optimal pyramid $\mathcal{P}$, the function $f_{\mathcal{P}}$ satisfies the following three conditions. Moreover, a function satisfying them gives the boundary surface of an optimal pyramid.*
(1) $f_{\mathcal{P}}$ is boundary surface of a pyramid.
(2) Each flat F of the above pyramid is a balancing flat; Consequently, $\int_{x \in F} f_{\mathcal{P}}(x)\mu(x)dx = \int_{x \in F} \rho(x)dx$.
(3) Among all functions satisfying (1) and (2), $f_{\mathcal{P}}$ maximizes the potential $\int_{x \in C}(f_{\mathcal{P}}(x))^2\mu(x)dx$.

Proof: The first condition is trivial, and we will show $f_{\mathcal{P}}$ satisfies (2) and (3). In each flat of height t of the optimal pyramid, the two regions $P = P_{t-\epsilon}$ and $P' = P_{t+\epsilon}$ for an infinitesimally small ϵ should have the same gain at t, and the flat must be $P' \setminus P$; hence the flat must be balanced, and (2) holds.

At each flat F with height t_F, the potential $A(F) = \int_{x \in F}(f_{\mathcal{P}}(x)^2\mu(x))dx$ within R is $(t_F)^2\mu(F)$. On the other hand, since F is balancing, $\rho(F) = t \cdot \mu(F)$; hence, $B(F) = \int_{t < t_F}(\rho(F) - t \cdot \mu(F))dt = t_F \cdot \rho(F) - \frac{t_F^2}{2} \cdot \mu(F) = \frac{t_F^2}{2} \cdot \mu(F)$. Therefore, $A(F) = 2B(F)$. This gives (3), since the sum of $B(F)$ over all flats equals $V(\mathcal{P})$, while that of $A(F)$ gives the potential defined in (3).

From the above proof, we can also see that a function satisfying the three conditions give the boundary surface of an optimal pyramid, since otherwise there exists a pyramid with a larger value of $V(\mathcal{P})$, and thus a larger potential; a contradiction. □

Consider a special case where $\mu \equiv 1$. Then, (2) of Lemma 2.1 says that we obtain a flat of $f_{\mathcal{P}}$ by leveling ρ locally, and (3) means that loss of potential is minimized. Indeed, if $d = 2$, we can consider ρ and $f_{\mathcal{P}}$ as surfaces of a terrain and its corresponding pyramid, and their potential (after normalization) corresponds to the positional potential of the terrain and the pyramid assuming that the gravity is a constant. Thus, we have the following intuitive description of the optimal pyramid (we use terminology for $d = 2$, but it can be generalized to any dimensional case):
If $\mu \equiv 1$, regarding ρ as a terrain, we have the optimal pyramid by moving earth from higher cells to lower cells with the minimum loss of positional potential to form a unimodal terrain such that each horizontal section belongs to $\mathcal{F}$.

2.2 Closed family

Let R_t^{opt} be the region in $\mathcal{F}$ maximizing $g_t(R; \rho, \mu)$. Intuitively, if t increases, R_t^{opt} is shrunk. If $\{R_t^{opt}\}$ forms a pyramid , it is obviously the optimal pyramid. Unfortunately, it is not always true that it is a pyramid; however, it holds if $\mathcal{F}$ satisfies certain conditions.

A (discrete) family $\mathcal{F}$ of regions in $\mathcal{R}^d$ is called a *closed family* [1] if it is closed under intersection and union operations, that is, $R \cap R' \in \mathcal{F}$ and $R \cup R' \in \mathcal{F}$ for every pair R, R' of regions in $\mathcal{F}$.

Proposition 2.2 *If $\mathcal{F}$ is a closed family, and let P_t^{opt} be the region in $\mathcal{F}$ maximizing $g_t(R, \rho, \mu)$. If there are more than one regions maximizing $g_t(R, \rho, \mu)$, we take any one which is minimal under inclusion. Then, $\{P_t^{opt}\}_{t \geq 0}$ gives the optimal pyramid $\mathcal{P}$.*

Proof: It suffices to show that $A = P_t^{opt} \subseteq B = P_{t'}^{opt}$ if $t > t'$. Note that the function $g_t(R, \rho, \mu) - g_t(R', \rho, \mu)$ is a nonincreasing function in t if $R' \subset R$. If $A \setminus B$ is not empty, $0 \geq g_{t'}(A \cup B, \rho, \mu) - g_{t'}(B, \rho, \mu) = g_{t'}(A, \rho, \mu) - g_{t'}(A \cap B, \rho, \mu) \geq g_t(A, \rho, \mu) - g_t(A \cap B, \rho, \mu) \geq 0$. Since $A = P_t^{opt}$, $g_t(A, \rho, \mu) \leq g_t(A \cap B, \rho, \mu)$; thus, $g_t(A, \rho, \mu) = g_t(A \cap B, \rho, \mu)$

[1] "Closed family" is not a common naming: it is a discrete analogue of "a system of closed sets" in topology theory.

and because of minimality under inclusion, $A = A \cap B$. Thus, $A \subseteq B$. □

Thus, if a family $\mathcal{F}$ is constructed by using closed families as its building blocks, we have hope to design an efficient algorithm for computing the optimal pyramid for $\mathcal{F}$.

3 One-dimensional problem

In this section, we consider one dimensional case where $\mathcal{F}$ is the set of all integral intervals. We omit most of proofs because of space limitation. The main result in this section is an $O(n \log n)$ time algorithm for computing the optimal pyramid. The algorithm is based on the fact that the family of all integral subintervals in $(0, n]$ containing a fixed element i is a closed family. Note that the family of all integral subintervals in $(0, n]$ is not a closed family, since a union of two nonintersecting intervals is not an interval. The rightmost index of an interval giving the top flat of a pyramid $\mathcal{P}$ is called the *peak index* of $\mathcal{P}$. The following lemma is trivial:

Lemma 3.1 *If i is the peak index of the optimal pyramid, the height of the pyramid is $\rho(i)/\mu(i)$.*

An index i is called *effective* if there is no interval J containing i such that $\int_{x \in J}(\rho(x)\mu(i) - \rho(i)\mu(x))dx > 0$. A peak index of an optimal pyramid must be effective. For a given effective index i, a pyramid $\mathcal{P}$ is called locally optimal if it maximizes $V(\mathcal{P})$ under the condition that its top flat contains i: the locally optimal pyramid is denoted by $\mathcal{P}(i)$.

We first show that the local optimal pyramid $\mathcal{P}(i)$ with the fixed peak index i can be computed in linear time. Then, using the ideas in the algorithm design, we proceed to an $O(n \log n)$ time algorithm for computing the (global) optimal pyramid.

3.1 Linear time algorithm for computing a local optimal pyramid

In this subsection, we fix an effective index i, and consider the local optimal pyramid $\mathcal{P}(i)$. It is easy to see that each flat of a local optimal pyramid must be a balancing flat.

Definition 3.2 *For a value t, $left_i(t)$ is the index $j \leq i$ minimizing $\int_{x \in (0,j]}(\rho(x) - t \cdot \mu(x))dx$. Similarly, $right_i(t)$ is the index $j \geq i$ minimizing $\int_{x \in (j,n]}(\rho(x) - t \cdot \mu(x))dx$.*

A value t is left-critical if $left_i(t)$ is not unique, that is, there exist two indices j_1 and j_2 ($j_1 < j_2 \leq i$) such that both of them minimize $\int_{x \in (0,j]}(\rho(x) - t \cdot \mu(x))dx$. Consequently, the interval $(j_1, j_2]$ becomes a balancing interval for t. Similarly, we define a right-critical value of t.

The following lemma is a direct consequence of Proposition 2.2 and Lemma 2.1.

Lemma 3.3 *The horizontal section of the pyramid $\mathcal{P}(i)$ at a height t is $(left_i(t), right_i(t))$, and it's exposed part is a flat if and only if t is critical.*

Thus, $left_i(t)$ and $right_i(t)$ are called left-end and right-end of the pyramid $\mathcal{P}(i)$ at t, respectively. We next examine the set of left-critical values. For an index j, we consider a planar point $p(j) = (m(j), u(j))$ where $u(j) = \int_{x \in (0,j]} \rho(x)dx$ and $m(j) = \int_{x \in (0,j]} \mu(x)dx$. Let $C^\ell(i)$ be the lower convex hull of the points $\{p(0), p(1), \ldots, p(i)\}$. Here, lower convex hull is the lower boundary chain of the convex hull of the points; thus, the lower convex hull is a list of vertices and edge segments on the chain. Similarly, $C^r(i)$ is the lower convex hull of $\{q(i+1), q(i+2), \ldots, q(n)\}$, where the points $q(j) = (m'(j), v(j))$ is defined by $m'(j) = \int_{x \in (j,n]} \mu(x)dx$. $v(j) = \int_{x \in (j,n]} \rho(x)dx$. Since they are convex hulls of sorted point sets, they can be computed in linear time [8]. It is routine to obtain the following lemma:

Lemma 3.4 *A value t is a left (resp. right) critical value if and only if t is a slope of an edge of $C^\ell(i)$ (resp. $C^r(i)$).*

The number of critical values is at most n, and they can be computed in $O(n)$ time. Thus, we can compute a local optimal pyramid in linear time.

3.2 An $O(n \log n)$ time algorithm

The algorithm in the previous subsection computes the local optimal pyramid for each effective index independently, and naively yields an $O(n^2)$ time algorithm for computing the global optimal pyramid.

Here, we give an improved method in which we grow the locally optimal pyramids simultaneously from the top, and weed out pyramids that are disclosed to be impossible to be the optimal pyramid during the process. If the optimal pyramid is not unique, we compute the one with the leftmost peak index. For a pyramid P and an interval I of x, the potential of P within I is $\int_I f_{\mathcal{P}}(x)^2\mu(x)dx$.

We consider a list L which is initially the sorted list of all effective indices. Let $next(i)$ be the next index to i in L. We remark that an index i is an effective index if and only if $p(i)$ a vertex of $C^\ell(i+1)$ and $q(i)$ is on the chain $C^r(i-1)$. The above conditions can be checked by running a linear time incremental convex hull algorithm (on sorting points) for each of two lower convex hulls $C^\ell(n)$ and $C^r(1)$. Thus, the initial list can be obtained in $O(n)$ time.

Definition 3.5 *For two indices i and j, their left meeting height is $t(left, i, j) = \max\{t : left_i(t) = left_j(t)\}$ and right meeting height is $t(right, i, j) = \max\{t : right_i(t) = right_j(t)\}$.*

Lemma 3.6 *In any sorted list L of effective indices, the following holds:*
(1) $left_i(t) = left_{next(i)}(t)$ if $t \le t(left; i, next(i))$.
(2) $right_i(t) = right_{next(i)}(t)$ if $t \le t(right, i, next(i))$.
(3) $t(left; i, j) \le t(left; i, next(i))$ and $t(right; i, j) \le t(right; i, next(i))$ for $i < j \in L$.

Therefore, the left end of the pyramid $\mathcal{P}(i)$ at the height t coincides with that of $\mathcal{P}(next(i))$ if t is below their left meeting height. Thus, $f_{\mathcal{P}(next(i))}(x) = f_{\mathcal{P}(i)}(x)$ if $x \le l(i) = left_i(t(left; i, next(i)))$ or $x \ge r(i) = right_i(t(right; i, next(i)))$.

We define the left-difference between the potential functions of $\mathcal{P}(i)$ and $\mathcal{P}(next(i))$ by $D_{left}(i, next(i)) = \int_{l(i)}^{next(i)} f_{\mathcal{P}(next(i))}(x)^2\mu(x)dx - \int_{l(i)}^{i} f_{\mathcal{P}(i)}(x)^2\mu(x)dx$.

Similarly, the right-difference is $D_{right}(i, next(i)) = \int_{next(i)}^{r(i)} f_{\mathcal{P}(next(i))}(x)^2\mu(x)dx - \int_{i}^{r(i)} f_{\mathcal{P}(i)}(x)^2\mu(x)dx$.

Definition 3.7 *If $D_{left}(i, next(i)) + D_{right}(i, next(i)) > 0$, i is called loser; otherwise, $next(i)$ is called loser.*

Lemma 3.8 *If i is a loser in any sorted list L of effective indices, $\mathcal{P}(i)$ is not the optimal pyramid.*

Our basic strategy is the following: Initially, $t = \infty$ and L is the list of all effective indices. We decrement t, and whenever we find an index i satisfying $t < \min\{t(left; i, next(i)), t(right; i, next(i))\}$, we judge which of i and $next(i)$ is a loser, and remove it from L. At last, L becomes a singleton list, the remaining index gives the peak of the optimal pyramid.

We need to decide $t < t(left; i, next(i))$ and $t < t(right; i, next(i))$ for each i, and also find losers effectively. Indeed, we can show that these operations can be done in $O(1)$ amortized time. Our basic tool is the *lower convex hull tree* (see e.g. [3]). Consider $T^\ell = \cup_{i \in L} C^\ell(i)$, which is a union of chains. Similarly, $T^r = \cup_{i \in L} C^r(i)$. They can be constructed in linear time.

For each edge e of the trees, we prepare an interval $I^l(e)$ (resp. $I^r(e)$) of indices i such that e is the convex hull of on $C^\ell(i)$ (resp. $C^r(i)$) for $i \in I^\ell(e)$ (resp. $I^r(e)$). For each slope t, consider the tangents of slope t to T^ℓ, and consider $I^\ell(e)$ for the left incident edge e to each tangent point. Then, it classify effective indices into groups, which we call left-grouping at t. Similarly, we consider right-grouping at t.

The value $t(left; i, j)$ is the slope of the edge at the branching point of $C^\ell(i)$ and $C^\ell(j)$. Similarly, the value $t(right; i, j)$ is the slope of the edge at the branching point of $C^r(i)$ and $C^r(j)$.

In the algorithm, we keep track of edges with slope t from leaves of the tree, and if we find a branching point of one of the trees T^ℓ and T^r, we eliminate one of the two chains meeting there. We keep the value $D_{left}(i, next(i))$ (resp. $D_{right}(i, next)$

) if both i and $next(i)$ are in the same left (resp. right) grouping. A new value of D_{left} (resp. D_{right}) is computed when we comes to the slope at a branching point of T^{ℓ} (resp. T^{r}). The decision of loser is done by using the formula in Definition 4.

Theorem 3.9 *The one-dimensional optimal pyramid can be computed in $O(n \log n)$ time.*

4 Higher-dimensional cases

In a higher dimensional case, the time complexity highly depends on the family $\mathcal{F}$ of regions. Indeed, it is not difficult to see that the problem is NP-hard for some families.

4.1 Family with a small number of regions

If $\mathcal{F}$ has M different regions, the optimal pyramid can be always computed in polynomial time in M and $N = n^d$ for the d-dimensional case. We construct a directed acyclic graph $H(\mathcal{F}) = (\mathcal{F}, E)$ whose vertex set is $\mathcal{F}$. For each pair R and R' of $\mathcal{F}$ we give a direct edge $e = (R, R')$ if and only if $R \supset R'$. We compute $t(e)$ satisfying that $\rho(R \setminus R') = t(e)\mu(R \setminus R')$. The value $t(e)$ is called the height label of e, and $r(e) = t(e)^2\rho(R \setminus R')/2$ is called the *profit* of e. A direct path $p = e_0, e_1, \ldots, e_q$ is called *admissible* if $t(e_{i-1}) < t(e_i)$ for $i = 1, 2, \ldots, q$. The profit of an admissible direct path is the sum of profit values of the edges in it.

Lemma 4.1 *The optimal pyramid is associated with the admissible path with the maximum profit in $H(\mathcal{F})$, such that $R \setminus R'$ is a flat of the pyramid if and only if (R, R') is an edge on the path.*

Thus, we can reduce the problem into a maximum-weight-path problem in the directed acyclic graph $H(\mathcal{F})$. Note that each path has at most N edges. By considering a dynamic programming algorithm, we obtain the following:

Theorem 4.2 *The optimal pyramid for $\mathcal{F}$ can be computed in $O(M^2N)$ time.*

However, we note that the above algorithm is seldom practical: for example, the family of rectangular regions has $O(N^2) = O(n^4)$ regions, and hence the above time complexity is $O(N^5) = O(n^{10})$. Moreover, M is often very large compared to N.

4.2 Two-dimensional closed families in a pixel grid

The method in the previous section is inefficient if M is large. Thus, we consider more efficient algorithms for some special families of regions.

Lemma 4.3 *Given a closed family $\mathcal{F}$, if the optimal pyramid has k flats and it takes $O(T)$ time for computing P_t^{opt} for any given t, we can compute the optimal pyramid in $O(kT)$ time.*

Proof: We can apply hand-probing method [1] to solve the problem. □

Although there are some families for which the computation of P_t^{opt} is NP-hard [1], there are several families with an exponential number of regions while the computation time T is polynomial in N. The number k of flats is at most N, and hence the time complexity for constructing the optimal pyramid is $O(NT)$; moreover, for particular cases, we can further improve the time complexity. We list such closed families in the two-dimensional pixel grid.

4.2.1 Connected lower half regions

A subregion R of the $n \times n$ pixel plane $\mathcal{G}$ is called a rectilinear lower half region (lower half region, in short) if there is a function $f_R(x)$ such that R is the union of pixels satisfying $y \leq f_R(x)$ holds in each of them. Unfortunately, a lower half region need not be connected, although connectedness of regions is essential in several applications. The family of connected lower half regions is one of region families adopted in SONAR data mining system [4, 7] in order to represent two-dimensional association rules The family of connected lower half regions is not a closed family, since the union of two nonintersecting connected lower half regions is not connected. However, given any index i, the

family of connected lower half regions having $(i,0)$ on their boundaries is a closed family, which we call connected i-lower half regions. Each column of a horizontal section of an optimal pyramid at t is an optimal nonempty prefix that mazimizes the parametric gain (it may be negative). There are $O(N)$ critical values of t at which the optimal prefix changes. We can compute the sorted set S of these critical values and preprocess the columns in $O(N \log N)$ time, so that the nonempty optimal prefix of each coloumn at t can be queried in $O(\log n)$ time. The optimal pyramid for the family of connected i-lower half regions for a fixed i can be computed in $O(N)$ time, assuming we have the sorted set S. Since there are $n = \sqrt{N}$ candidates of i, we can compute the optimal pyramid for the family of all connected lower half regions in $O(N^{3/2})$ time. Furthermore, we can improve the time complexity as follows:

Theorem 4.4 *The optimal pyramid for the family of all connected lower half regions can be computed in $O(N \log^2 N)$ time.*

4.2.2 Downstep regions and stabbed unions

In many applications, we want to consider a family of regions whose boundary curves have nice shapes. A subregion R of $\mathcal{G}$ is called a downstep region if there is a nonincreasing function $f_R(x)$ such that R is the union of pixels satisfying $y \leq f_R(x)$ in each of them. There are exactly 2^n members in the family.

Lemma 4.5 *The optimal region P_t^{opt} for the family of downstep regions can be computed in $O(N)$ time for each t.*

Proof: We first compute $\gamma(i,j) = \sum_{s=1}^{i} \rho((s,j)) - t \cdot \mu((s,j))$, which is the gain in the first i pixels in the j-th column. This can be done in $O(N) = O(n^2)$ total time for all (i,j). We sweep the pixel grid $\mathcal{G}$ from left to right, and compute $\alpha(i,j)$ which is the maximum gain of a downstep region up to the j-th column such that the region contains (i,j)-th pixel as the top pixel at the j-th column. Then it is easy to see that $\alpha(i,j) = \max\{\alpha(i,j-1) + \gamma(i,j), \alpha(i+1,j) - \gamma(i+1,j) + \gamma(i,j)\}$. Since this computation is done in constant time for each (i,j), we can compute all of $\alpha(i,j)$ in $O(N)$ time by dynamic programming. The gain of the optimal region is $max_{i=0}^{n} \alpha(i,n)$, and the region can be computed by backtracking the dynamic programming process. □

Thus, we can compute the optimal pyramid in $O(N^2)$ time by naively applying Lemma 4.3. We can improve the time complexity as follows:

Lemma 4.6 *Suppose we have the optimal region $P_{t(1)}^{opt}$ and $P_{t(2)}^{opt}$ for a parameter value $t(1) < t(2)$, and $P_{t(1)}^{opt} \setminus P_{t(2)}^{opt}$ has Y pixels. Then, for any t satisfying $t(1) < t < t(2)$, P_t^{opt} can be computed in $O(Y)$ time. Here, we output the boundary curve of P_t^{opt} by using that of $P_{t(2)}^{opt}$ and indicating how to update it.*

Proof: It suffices to consider each connected component of $P_{t(1)}^{opt} \setminus P_{t(2)}^{opt}$ in the pixel grid $\mathcal{G}$ separately. We can modify the dynamic programming algorithm given in the previous lemma for processing within each component in linear time with respect to the number of pixels in it. □

Theorem 4.7 *Suppose that the values of ρ and μ are quotient numbers of integers less than Γ. The optimal pyramid for the family of downstep regions can be computed in $O(\min\{N \log N \log(N\Gamma), N^{3/2} \log^2 N\})$ time.*

Proof: For a pair $t(1) < t(2)$ of parameter values such that $|P_{t(1)}^{opt} \setminus P_{t(2)}^{opt}| = Y$, a critical separator t is the parameter value satisfying that $t(1) < t < t(2)$, $|P_t^{opt} \setminus P_{t(2)}^{opt}| < 3Y/4$ and for any $t' > t$ $Y/4 < |P_{t'}^{opt} \setminus P_{t(2)}^{opt}|$. Note that if we can find t satisfying that$Y/4 < |P_t^{opt} \setminus P_{t(2)}^{opt}| < 3Y/4$, it is a critical separator, although such a t may not always exist. It is easy to see that if we can compute the critical separator in $f(Y)$ time satisfying that $f(Y) = \Omega(n)$, then we attain $O(f(N) \log N)$ time to compute the optimal pyramid.

Indeed, we can compute a critical separator by using binary search in $O(Y \log(N\Gamma))$. If we

want to design an algorithm independent of Γ, we can apply parametric searching to replace $\log(N\Gamma)$ with $N^{1/2}\log N$ in the time complexity by using a parallel-structured algorithm with $O(N^{1/2}\log N)$ steps for its guide algorithm (we omit routine details). □

We fix a grid point $p = (x_p, y_p)$ of $\mathcal{G}$. A region R is called *stabbed union* (of rectangles) at p if it is a union of rectangles each of which contains p in its closure. This is two-dimensional analogue of the family of intervals including a given index i discussed in Section 3.1, and it is easy to see that the family is closed. It is a restricted class of rectilinear convex regions, such that if we cut a stabbed union with lines $x = x_p$ and $y = y_p$, we have four (reflected) downstep regions, indeed, the family of stabbed union at p is the closure (under union and intersection) of the family of rectangles containing p.

A discretized rhombus (whose diagonals are axis-parallel) is a stabbed union, thus a horizontal section of a real pyramid (rotated by 45 degrees) is a stabbed union. Moreover, discretization of an axis parallel elliptic region is a stabbed union. The pyramid Figure 2 is optimal with respect to the family of all stabbed unions We have the following theorem as a corollary of Theorem 4.7:

Theorem 4.8 *The optimal pyramid for the family of stabbed unions at p can be computed in $O(\min\{N\log n\log\Gamma, N^{3/2}\log n\})$ time.*

5 Concluding Remarks

We have given an $O(n\log n)$ algorithm for one-dimensional pyramid construction; however, we hope that a linear time algorithm exists. For two-dimensional cases, we have strong restriction for the families of regions in order to design efficient algorithms. It is open whether it is NP hard or not if we consider the family of rectilinear convex regions or that of x-monotone regions. Moreover, it is interesting question to improve the time complexity for the family of rectangles. We plan to implement some of algorithms to apply them to construct a flexible (but optimized) data mining system by using pyramids for giving decision rules that can automatically avoid unnecessary classification.

References

[1] T. Asano, D. Chen, N. Katoh, and T. Tokuyama, Efficient Algorithms for Optimization-Based Image Segmentation, *International Journal of Computational Geometry and Applications* **11** (2001) 145-166.

[2] I. Bloch, Spatial Relationship between Objects and Fuzzy Objects using Mathematical Morphology, in *Geometry, Morphology and Computational Imaging*, 11th Dagsthul Workshop on Theoretical Foundations of Computer Vision, April 2002.

[3] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama, Mining Optimized Association Rules for Numeric Attributes, *Journal of Computer and System Sciences* **58** (1999) 1-12.

[4] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama, Data Mining with Optimized Two-Dimensional Association Rules, *ACM Transaction of Database Systems* **26** (2001) 179-213.

[5] H. Gabow and R. Tarjan, A Linear Time Algorithm for a Special Case of Disjoint Set Union, *Journal of Computer Systems and Sciences* **30** (1985) 209-221.

[6] Y. Morimoto, H. Ishii and S. Morishita, Construction of Regression Trees with Range and Region Splitting, *The 23rd VLDB Conference* (1997) 166-175.

[7] Y. Morimoto, T. Fukuda, S. Morishita, and T. Tokuyama, Implementation and Evaluation of Decision Trees with Range and Region Splitting, *Constraints* (1997) 402-427.

[8] F. P. Preparata and M. I. Shamos, *Computational Geometry – An Introduction*, Springer-Verlag, 1988 (2nd ed.).