1C3-1

木構造データからの主成分抽出

Extracting Principal Component from Tree Structure Data

山崎朋哉 *1 Tomoya Yamazaki　山本章博 *1 Akihiro Yamamoto　久保山哲二 *2 Tetsuji Kuboyama

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

*2学習院大学 計算機センター Computer Centre, Gakushuin University

We propose two new methods for extracting principal components from a set of rooted labeled trees by generalizing the notion of principal components in conventional statistics. Our methods, top-down and bottom-up methods, extend the earlier work for binary trees and restricted unlabeled rooted trees. The top-down method proposed in this paper is an extension of these previous researches so that it can be applied to labeled rooted unordered trees. The bottom-up method is for extracting principal component subtrees, not paths, but is formulated in an analogous way. Both of the proposed methods run in linear time. We confirmed the validity of our method by applying to glycan data.

1. はじめに

データの集合を互いに相関の無い特徴にまとめることは, データマイニングにおいて重要な問題である. その手法として高次元数値ベクトルデータに対しては, Pearson [8] によって提唱された主成分分析 (以下 PCA) がある. PCA は, 高次元データをできるだけ情報量の損失の少ない低次元空間へ射影する. 本稿では, 木構造データに主成分分析を考察する. 木構造データ全体からなる木空間はユークリッド空間ではないため, 一般的な PCA を単に木構造データに適用することはできない.

木構造データに対する PCA は近年研究が進んできている. Wang ら [4] によって初めて木構造に対する PCA が定式化され, MRA 画像から得られた血管の構造を表す二分木に対して適用された. そして, Aydin ら [2] によってラベル無し根付き二分木に対する PCA を線形時間で解くアルゴリズムが提唱され, Alfaro ら [3] は, Aydin らの手法をラベル無し根付き順序木 (ただし, インデックスが固定されるという強い制約がある) に拡張した. これらの既存手法では, 入力データ上の空間を入力データ中の木全ての和集合である support tree とし, 各入力データの射影先の空間を support tree 上の tree-line と定義している. Tree-line とは, support tree 上の任意の木 l_0 が与えられたとき, $\{l_0, \ldots, l_k\}$ と定義され, l_i は l_{i-1} から一方向に子ノードを付与したものである. ゆえに, 木空間を構成する一次元の軸の模倣として tree-line を, 原点として l_0 を定義することで, ユークリッド空間に似せた木空間を定義することができる. また, 全空間を support tree ではなく, ノードや枝の操作を軸にした空間に射影し, その空間での測地線を主成分とする研究も進められている [1, 11].

本稿ではまず Alfaro らの問題を, より一般的な木であるラベル付き根付き無順序木へ拡張する. その木に対して主成分がパス (tree-line) であるトップダウン手法と主成分が部分木であるボトムアップ手法の二種類の主成分の抽出方法を提案する. さらに, 提案手法が Alfaro らのアルゴリズムと同様に線形時間計算量であることを示す. 実際に提案した手法を糖鎖データに適用し, 複数のデータで分類精度を測り, また各主成分がどの程度入力データを表現するかを求めることで妥当性を示した.

連絡先: 山崎朋哉
所属：京都大学大学院情報学研究科
〒 606-8501 京都府京都市左京区吉田本町総合研究 7 号館 3 階 324・327 号室
E-mail：t.yamazaki@iip.ist.i.kyoto-u.ac.jp

2. 準備

本稿で扱う木とは, 非巡回連結有向グラフであり, V をノード集合, $E \subseteq V \times V$ を枝集合としたとき, 木は $t = (V, E)$ と表す. ノード間の祖先関係を $\leq$ で表し, $x, y \in V$ に対して, $x \leq y$ は y が x の祖先であることを表す. 特に $x \leq y$ かつ $x \neq y$ のとき, $x < y$ と記す. 根ノードでないノード $v \in V$ の親ノードを $parent(v)$ と記す.

2.1 マッピング

二つの木 $t_1 = (V_1, E_1), t_2 = (V_2, E_2)$ に対して, ノード $x \in V_1$ の削除と挿入, ノード $x \in V_1$ のラベルを別のノード $y \in V_2$ のラベルへ置換する 3 つの操作を編集操作と呼び, それぞれの編集コストを $\gamma(x \to \lambda), \gamma(\lambda \to x), \gamma(x \to y)$ と記す. これら全てのコストが等しいとき, 木 t_1 から木 t_2 に変換するための編集操作に要するコストの最小値を編集距離と呼ぶ. また, ノードの削除と挿入のコストが 1, ラベルの置換のコストが 2 であるとき, 編集操作にラベルの置換を考慮しないことと同じである. このとき, 木 t_1 から木 t_2 に変換するための編集操作に要するコストの最小値を indel(insert-deletion) 距離と呼ぶ.

マッピング $M \subseteq V_1 \times V_2$ とは, 二つの木構造間のノードの対の集合であり, それに与える制約によって様々なマッピングが存在する. また, $M|^{t_1}, M|^{t_2}$ をそれぞれ t_1, t_2 中で M に含まれるノードの集合とし,

$$M|^{t_1} = \{x_1 \in V_1 \mid \exists x_2 \in V_2, (x_1, x_2) \in M\}$$
$$M|^{t_2} = \{x_2 \in V_2 \mid \exists x_1 \in V_1, (x_1, x_2) \in M\}$$

と定義する. このとき, 編集に要するコストが, 以下の式で与えられる.

$$\gamma(M) = \sum_{(x,y)\in M} \gamma(x \to y) + \sum_{x \in M|^{t_1}} \gamma(x \to \lambda) + \sum_{y \in M|^{t_2}} \gamma(\lambda \to y)$$

また, t_1 と t_2 の間の編集距離を求めることは $\gamma(M)$ を最小にするようなマッピング M を発見することと等価である [9].

Tai マッピング [9]

二つの木が順序木であるときの Tai マッピング M の条件を以下に示す. マッピング M に対して, 任意のノードの対 $(x_1, x_2), (y_1, y_2) \in M$ が以下の条件を満たすとき, M は Tai マッピングであるという.

$$x_1 = y_1 \Longleftrightarrow x_2 = y_2 \text{ (一対一の関係)},$$
$$x_1 < y_1 \Longleftrightarrow x_2 < y_2 \text{ (祖先関係の保存)},$$
$$x_1 \preceq y_1 \Longleftrightarrow x_2 \preceq y_2 \text{ (兄弟関係の保存)}.$$

ここで, $\preceq$ は兄弟間にあるノード間の順序関係を表す. ただし, 無順序木の場合, 兄弟関係は考慮しないため, 3 番目の条件は考慮しない.

トップダウンマッピング [6]

M が木 t_1, t_2 の Tai マッピングであるとき, 任意のノードの対 $(s,t) \in M$(ただし, s,t は根ノードではない) に対して, $(parent(s), parent(t)) \in M$ が存在するとき, M をトップダウンマッピングという.

ボトムアップマッピング [5]

木 $t = (V, E)$ 中の $v \in V$ を根とする完全部分木を $t(v)$ とおき, 木 t_1, t_2 の Tai マッピングを M とおくとき, 任意のノードの対 $(s,t) \in M$ が以下の 2 つの条件

$$\forall s' \in t_1(s) \text{ s.t. } \exists t' \in t_2(t), (s', t') \in M,$$
$$\forall t' \in t_2(t) \text{ s.t. } \exists s' \in t_1(s), (s', t') \in M,$$

を満たすとき, M をボトムアップマッピングという.

2.2 既存手法

本節では, Alfaro ら [3] により定式化された木構造の PCA について説明する. 本節で扱う木はラベル無し順序木 (以下木と呼ぶ) である. 入力データである木 $t_i (i = 1, \ldots, n)$ に対して, $\mathcal{T} = \{t_1, \ldots, t_n\}$ と表す. 本節では, $\mathcal{T}$ 中の木は全てインデックスを共有すると仮定する. 木 t のノードのインデックスの集合を $Index(t)$, 木 t の葉のノードの集合を $Leaf(t)$ と記す. 入力データ $\mathcal{T}$ の support tree とは, 以下の式を満たす木 $ST(\mathcal{T})$ である.

$$Index(ST(\mathcal{T})) = \bigcup_{t \in \mathcal{T}} Index(t).$$

ノード $N \in V$ のインデックスが $i_1 i_2 \ldots i_n$ のとき, これは即ち根ノードから N までのパスを表す. また, 各ノードのインデックスは根からそのノードまでのパス上のノードのインデックスの集合としても扱う. 即ち, $i_1 i_2 \ldots i_n$ を $\{\epsilon, i_1, i_1 i_2, \ldots, i_1 i_2 \ldots i_n\}$ とみなす. Tree-line $TL(N)$ を

$$TL(N) = \{\epsilon, i_1, i_1 i_2, \ldots, i_1 i_2 \ldots i_n\}$$

と定義する. また, $TL(N)$ はパスの集合であるため, パスが成す木の集合とみなしてよい. 二つの木間の距離を,

$$d_{Tai}(t_1, t_2) = |Index(t_1) \setminus Index(t_2)| + |Index(t_2) \setminus Index(t_1)|$$

と定める. これは Tai マッピングに基づく indel 距離に等しいことが示せる [3]. 木 t の tree-line への射影を

$$P_N(t) = \underset{l \in TL(N)}{\arg\min} \{d_{Tai}(t, l)\}$$

と定義する. 以上を踏まえて, 第 1 主成分を表すパスを表すノードを,

$$N_1 = \underset{N \in Leaf(ST(\mathcal{T}))}{\arg\min} \sum_{t \in \mathcal{T}} d_{Tai}(t, P_N(t))$$

と定める. また, tree-line の和を以下のように定める.

$$TL(N_1) \uplus \cdots \uplus TL(N_k) = \{\{l_1\} \cup \cdots \cup \{l_k\} \mid l_1 \in TL(N_1), \ldots, l_k \in TL(N_k)\}$$

また, 木 t の $TL(N_1) \uplus \cdots \uplus TL(N_k)$ への射影を

$$P_{N_1 \cup \cdots \cup N_k} = \underset{l \in TL(N_1) \uplus \cdots \uplus TL(N_k)}{\arg\min} \{d_{Tai}(t, l)\}$$

と定める. このとき, 第 k 主成分を表すパスを表すノードは,

$$N_k = \underset{N \in Leaf(ST(\mathcal{T}))}{\arg\min} \sum_{t \in \mathcal{T}} d_{Tai}(t, P_{N_1 \cup \cdots \cup N_{K-1} \cup N}(t))$$

と定める.

3. 提案手法

本節では, 前節の木 (入力データ内で共通のインデックスを持つラベル無し根付き順序木) を一般的なラベル付き根付き無順序木に拡張する. 本節以降で扱う木 $t = (V, E)$ はラベル付き根付き無順序木とする. ラベルはアルファベット Σ 中の一つの記号であり, 各ノードに一つのラベルが付与される. 木 $t = (V, E)$ に対して, ラベル関数を $\alpha : V \to \Sigma$ と定義する. また, i が木 $t = (V, E)$ 中の $N \in V$ のインデックスであるとき, $\alpha(N)$ を $\alpha(i)$ と表す.

3.1 トップダウン手法

ノード $N \in V$ の木 T 上でのインデックスが $i_1 i_2 \ldots i_n$ であるとき, 根ノードからノード N までのパス上のラベル列を

$$Path(N) = \alpha(i_1)\alpha(i_1 i_2) \ldots, \alpha(i_1 \ldots i_n)$$

と定義する. また, 木 t の根ノードから各葉ノードまでのパス (ラベルの有限列) の集合は,

$$Fiber(t) = \{Path(N) \mid N \in Leaf(t)\}$$

と定義する. 前節では, 全空間を support tree として定めていたが, 本節では, 全空間を入力データ $\mathcal{T}$ の support fiber($SF(\mathcal{T})$) として以下のように定める.

$$SF(\mathcal{T}) = \bigcup_{t \in \mathcal{T}} Fiber(t)$$

パス p の tree-line は

$$TL(p) = \{\lambda\} \cup \bigcup_{n \in p} Path(n)$$

と定める. 二つの木間のトップダウンマッピングに基づく indel 距離を $d_{td}(t_1, t_2)$ として定める. $M_{td}^{t_1, t_2}$ は置換を考慮しないトップダウンマッピングに含まれるノード集合とする. 木 t_1, t_2 の根ノードからの完全一致部分のノードの対の集合が即ち $M_{td}^{t_1, t_2}$ である. また, 木 t の tree-line($TL(p)$) への射影は,

$$P_p(t) = \underset{l \in TL(p)}{\arg\min} \{d_{td}(t, l)\}$$

とおける．このとき，第 1 主成分を表すパスは，

$$P_1 = \arg\min_{p \in SF(\mathcal{T})} \sum_{t \in \mathcal{T}} d_{td}(t, P_p(t))$$

とおける．Tree-line の和は，前節と同様に

$$\begin{aligned} &TL(P_1) \uplus \cdots \uplus TL(P_k) \\ =&\{p_1 \cup \cdots \cup p_k \mid p_1 \in TL(P_1), \ldots, p_k \in TL(P_k)\} \end{aligned}$$

と定める．また，木 t の $TL(P_1) \uplus \cdots \uplus TL(P_k)$ への射影を

$$\begin{aligned} &P_{P_1 \cup \cdots \cup P_k}(t) \\ =&\arg\min_{PS \in TL(P_1) \uplus \cdots \uplus TL(P_k)} \sum_{p \in PS} d_{td}(t,p) + \sum_{\substack{p_i, p_j \in PS \\ p_i \neq p_j}} |M_{td}^{p_i, p_j}| \end{aligned}$$

とおく．以上のことから，第 k 主成分を表すパスは，

$$P_k = \arg\min_{P \in SF(\mathcal{T})} \sum_{t \in \mathcal{T}} \sum_{p \in P_{P_1 \cup \cdots \cup P_{k-1} \cup P}(t)} d_{td}(t,p) \qquad (1)$$

とおける．これは，前節で示した手法の拡張である．

3.2 ボトムアップ手法

木 t_1, t_2 のボトムアップマッピングにより得られる同型完全共通部分木の集合を $Iso(t_1, t_2)$ と表す．このとき，入力データ $\mathcal{T} = \{t_1, \ldots, t_n\}$ に対して，全空間を $SI(\mathcal{T})$ として以下のように定める．

$$SI(\mathcal{T}) = \{Iso(t_i, t_j) \mid i, j \in \{1, \ldots, n\}, i \neq j\}$$

木 t の全ての完全部分木の集合を $Sub(t)$ と定める．二つの木間のボトムアップマッピングによる indel 距離を $d_{bu}(t_1, t_2)$ として定める．また，$M_{bu}^{t_1,t_2} \subseteq V_1 \times V_2$ は木 $t_i = (V_i, E_i)(i = 1, 2)$ 間の置換を考慮しないボトムアップマッピングに含まれるノード集合とする．このとき，木 t の射影先の空間を表す木を S とするとき，t の射影は，

$$P_S(t) = \arg\min_{s \in Sub(S)} \{d_{bu}(t, s)\}$$

と定義する．このとき，第 1 主成分は，

$$S_1 = \arg\min_{S \in \mathcal{ST}(\mathcal{T})} \sum_{t \in \mathcal{T}} d_{bu}(t, P_S(t))$$

また，$S_1 \ldots S_k$ を木としたとき，それらの $Sub(S_i)(i = 1, \ldots, k)$ の和は，

$$\begin{aligned} &Sub(S_1) \uplus \cdots \uplus Sub(S_k) \\ =&\{s_1 \cup \cdots \cup s_k \mid s_1 \in Sub(S_1), \ldots, s_k \in Sub(S_k)\} \end{aligned}$$

と定義する．また，前節と同様に，木 t の $Sub(S_1) \cup \cdots \cup Sub(S_k)$ への射影を

$$\begin{aligned} &P_{S_1 \cup \cdots \cup S_k}(t) \\ =&\arg\min_{SS \in Sub(S_1) \uplus \cdots \uplus Sub(S_k)} \sum_{s \in SS} d_{bu}(t,s) + \sum_{\substack{s_i, s_j \in SS \\ s_i \neq s_j}} |M_{bu}^{s_i, s_j}| \end{aligned}$$

とおく．以上のことから，第 k 主成分を表す部分木は，

$$S_k = \arg\min_{S \in SI(\mathcal{T})} \sum_{t \in \mathcal{T}} \sum_{s \in P_{S_1 \cup \cdots \cup S_{k-1} \cup S}(t)} d_{bu}(t,s)$$

とおける．

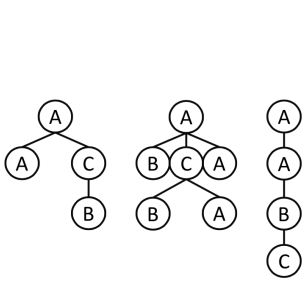

図 1: 入力データの例

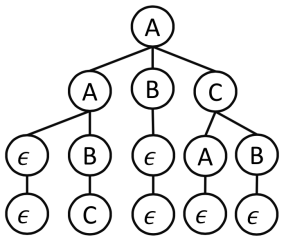

図 2: 入力データに対する super tree．ここで ϵ ノードは終端ノードを表し，全パスの深さを揃えるために生成する．

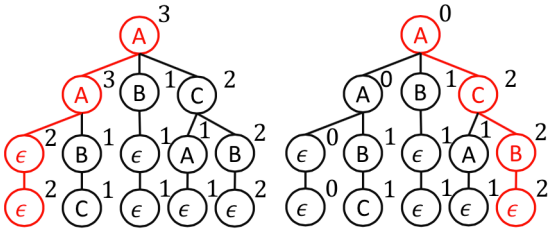

図 3: 各ノードの右肩の数字はノードの重みを表す．左図と右図の赤線部がそれぞれ第 1 主成分と第 2 主成分のパスを表す．

3.3 提案手法の解釈

トップダウン手法において，$SF(\mathcal{T})$ の各パスのラベルをインデックスとみる super tree [*1] を作る．このとき，各パスに対して，$\epsilon(\notin \Sigma)$ ラベル [*2] を持つノードを全パスの深さが一致するまで加える．入力データの例を図 1 に，その入力データに対する super tree を図 2 に示す．

$\mathcal{T}$ の第 k 主成分は，木 t のノード集合を V_t，$\mathcal{T}$ の super tree を $Supt(\mathcal{T})$ として，$Supt(\mathcal{T})$ をパスの集合とみたとき，(1) 式から，

$$P_k = \arg\max_{P \in Supt(\mathcal{T})} \sum_{v \in P} \sum_{t \in \mathcal{T}} w_k(v, t, P)$$

を導くことができる．ただし，

$$w_k(v,t,P) = \begin{cases} 1 & v \in M_{td}^{P,t}|^P \text{and } v \notin P_1 \cup \cdots \cup P_{k-1}, \\ 0 & \text{o.w.} \end{cases} \qquad (2)$$

である．このことから，第 k 主成分は super tree の各ノードに (2) 式の重みを与え，その重みの合計が最大となるパスを求めることに等しい．図 1 の第 1 主成分，第 2 主成分を表すパスを図 3 に示す．また，得られた super tree のインデックスを入力データの木が共有することで，Alfaro ら [3] と同様に線形時間計算量で主成分が求められることが示せる．

3.4 実験

実際に KEGG/GLYCAN データベース [7] から取得した糖鎖のデータセットを用いて実験を行った．また，得られたデータセット中のデータは Leukemic, Erythrocyte, Serum, Plasma のいずれかのクラスに分類されている．表 1 に用いたデータ

*1 入力データの根ノードのラベルが異なる場合は，super tree の根ノードとしてダミーノードを与える．

*2 図 2 において $\epsilon \notin \Sigma$ ラベル付きのノードを生成する理由は，AA というパスを super tree 上に表すためである．

クラス	Leukemic	Erythrocyte	Serum	Plasma
データ数	140	127	78	60

表 1: 糖鎖データのクラスと各クラスのデータ数

クラス	第 1 主成分
Leukemic	GlcNAc.-GlcNAc.1b4-Man.1b4-Man.1a6 -GlcNAc.1b2-Gal.1b4-NeuAc.2a3
Erythrocyte	AA.-Glc.1b1-Gal.1b4-GlcNAc.1b3 Gal.1b4-GlcNAc.1b3-Gal.1b4-GlcNAc.1b3 -Gal.1b4-Fuc.1a2
Serum	GlcNAc.-GlcNAc.1b4-Man.1b4-Man.1a3 -GlcNAc.1b2-Gal.1b4-NeuAc.2a6
Plasma	GlcNAc.-GlcNAc.1b4-Man.1b4-Man.1a6 -GlcNAc.1b2-Gal.1b4-Fuc.1a2

表 2: 各入力データのクラスとトップダウン手法によって抽出された第 1 主成分を表すパス

セットとそのデータ数を記す. 糖鎖データは各ノードと枝にラベルが付与されているが, 本稿では [10] と同様に, 枝のラベルをその枝の子ノードのラベルとみなす. そして, 本稿では, 糖鎖データを順序木ではなく, そのようにして得られたラベル付き根付き無順序木として扱う. このとき得られた各クラスの第 1 主成分を表 2 に記す. この結果の妥当性を評価するために, まず各クラスの入力データの第 1〜 第 5 主成分まで求める. 各主成分のパスの和集合である木を各クラスを表す木として, 全入力データを分類するときの精度を表 3 に示す. この結果から, 糖鎖データに対しては高い分類精度を示すことが分かる. また, 第 k 主成分まで求めたときの各クラスの入力データ中のノードの表現数を図 4 に示す. このことから, Leukemic が他のデータに比べて最も各主成分の表現力が高いことが分かる.

4. まとめ

本稿では, ラベル付き根付き無順序木から主成分を抽出する二種類の方法を提案した. トップダウン手法は糖鎖データのように枝分かれが少なく, ラベル数が多い木に対して有効で, ボトムアップ手法は枝分かれが多く, 葉ノードが重要な木に対して有効であると考えられる.

今後の課題として, 両手法の統合や, トップダウンマッピングとボトムアップマッピング以外のマッピングに基づく主成分抽出方法などが考えられる.

Leukemic	Erythrocyte	Serum	Plasma
0.900	0.811	0.679	0.850

表 3: 各クラスの分類精度

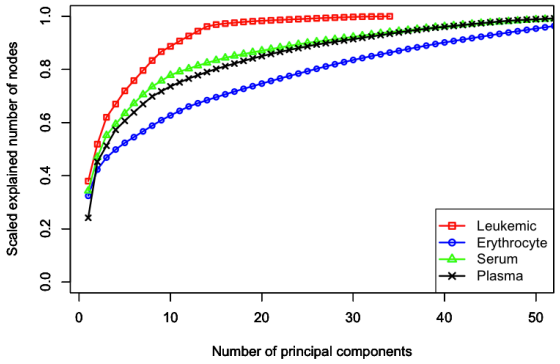

図 4: x 軸は各入力データの第 k 主成分. y 軸は第 1 〜 第 k 主成分で表現するノード数の割合.

参考文献

[1] A. Feragen, P. Lo, M. D. Bruijine, M. Nielsen, and F. Lauze : Toward a Theory of Statistical Tree-Shape Analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 8, pp. 2008-2021 (2013).

[2] B. Aydin, C. Pataki, H. Wang, E. Bullitt, and J. S. Marron : A Principal Component Analysis For Trees, *The Annuals of Applied Statistics*, Vol. 3, No. 4, pp. 1597-1615 (2009).

[3] C. A. Alfaro, B. Aydin, C E. Valencia, E. Bullitt, and A. Ladha : Dimension Reduction in Principal Component Analysis for Trees, *Computational Statistics and Data Analysis*, vol. 74, pp. 157-179 (2014).

[4] H. Wang and J. S. Marron : Object Oriented Data Analysis : Set of Trees, *The Annals of Statistics*, Vol. 35, No. 5, pp. 1849-1873 (2007).

[5] G. Valiente : An Efficient Bottom-up Distance between Trees, *Proc. of 8th International Symposium on String Processing and Information Retrieval (SPIRE), IEEE Computer Science Press*, pp.212-219, (2001).

[6] J. T. -L. Wang and K. Zhang : Finding Similar Consensus between Trees : an Algorithm and a Distance Hierarchy, *Pattern Recognition* Vol. 34, pp. 127-137 (2001).

[7] K. Hashimoto, S. Goto, S. Kawano, K. F. Aoki-Kinoshita, and N. Ueda,: KEGG as a Glycan Informatics Resource,*Glycobiology*, Vol. 16, pp. 6370 (2006)

[8] K. Pearson : On lines and planes of closest fit to systems of points in space, *Phil. Mag. 2* Vol. 6, pp 559-572 (1901).

[9] K. C. Tai : The tree-to-tree correction problem, *J. Assoc. Comput*, pp. 422-433 (1979).

[10] T. Kuboyama, K. Hirata, K. F. Aoki-Kinoshita, H. Kashima, and H. Yasuda : A Gram Distribution Kernel Applied to Glycan Classification and Motif Extraction, *Genome Informatics*, Vol. 17(2), pp. 25-34 (2006).

[11] T.M.W. Nye, Principal Components Analysis in the Space of Phylogenetic Trees, *Ann. Statistics*, Vol. 39, No. 5, pp. 2716-2739 (2011).

[12] Y. Yamanishi, F. Bach, and J. P. Vert : Glycan Classification with Tree Kernels, *Bioinformatics*, Vol. 23, No. 10, pp. 12111216 (2007).