情報処理学会論文誌　データベース　Vol.7　No.2　37–50 (June 2014)

大規模時系列データの特徴自動抽出

松原 靖子[1,a)]　櫻井 保志[1]　Christos Faloutsos[2]

受付日 2013年12月22日, 採録日 2014年2月4日

概要：本論文では，大規模時系列データのための特徴自動抽出手法である AutoPlait について述べる．AutoPlait は，様々な時系列パターンを含む複雑なシーケンスが与えられたときに，そのシーケンスデータの中から重要な特徴を発見し，それらの情報を統計的に要約，表現する．提案手法は，(a) 大規模時系列データの中から類似した部分シーケンスのパターンを抽出し，(b) 計算量は入力データのサイズに対して線形である．さらに，最も重要な点として，(c) 提案手法はパラメータに依存しない．すなわち，事前情報の付与またはパラメータのチューニングを行うことなく，大規模シーケンスのパターン発見と特徴抽出を自動で行うことができる．実データを用いた実験では，AutoPlait が様々な時系列データの中から有用なパターンを正確に発見することを確認し，さらに，最新の既存手法と比較を行い提案手法が大幅な精度，性能向上を達成していることを明らかにした．

キーワード：時系列データ，特徴自動抽出

Fully Automatic Mining of Large Time-series Datasets

Yasuko Matsubara[1,a)]　Yasushi Sakurai[1]　Christos Faloutsos[2]

Received: December 22, 2013, Accepted: February 4, 2014

Abstract: In this paper we present AutoPlait, a fully automatic mining algorithm for co-evolving time sequences. Our method has the following properties: (a) effectiveness: it operates on large collections of time-series, and finds similar segment groups that agree with human intuition; (b) scalability: it is linear with the input size, and thus scales up very well; and (c) AutoPlait is parameter-free, and requires no user intervention, no prior training, and no parameter tuning. Extensive experiments on *67 GB* of real datasets demonstrate that AutoPlait does indeed detect meaningful patterns correctly, and it outperforms state-of-the-art competitors as regards accuracy and speed: AutoPlait achieves near-perfect, over *95%* precision and recall, and it is up to *472 times* faster than its competitors.

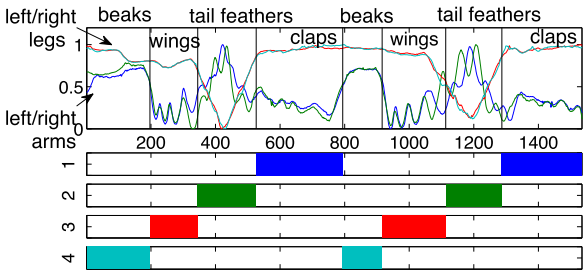
Keywords: time-series data, automatic mining

1. まえがき

時系列シーケンスは，センサデータや Web アクセス履歴等，様々なアプリケーションにおいて大量に生成されている．これらの大規模な時系列シーケンスの中から，典型的なパターンや異常値を発見することは非常に重要な課題である．本論文では，大規模時系列データを対象とし，重要な時系列パターンの抽出を自動的に行うことを目的とする．より具体的には，複数のシーケンスから構成される大規模時系列データ，つまり多次元時系列シーケンスを扱い，これらのデータ全体を表現する要約情報を抽出する．

一般に，実際に生成される時系列データは，複数の異なるトレンドやパターンを持つことが多い．たとえば，Web アクセス履歴のシーケンスは，平日と休日に異なるパターンを持つ．また，ネットワーク通信のモニタリングシステムから発生するデータについては，正常と異常のパターンが考えられる．ここで，これらの時系列パターンを本論文では「レジーム（regime）」と呼ぶ．本研究では，大規模時系列データの中から，これらの異なるトレンドを発見し，す

[1] 熊本大学
Kumamoto University, Kumamoto 860–8555, Japan
[2] カーネギーメロン大学
Carnegie Mellon University, Pittsburgh, PA, 15213–3891, USA
[a)] yasuko@cs.kumamoto-u.ac.jp



(a) AutoPlait の出力結果

(b)「チキンダンス（chicken dance）」における
4 つの代表的なステップ

図 1　*MoCap* データにおける AutoPlait の出力例

Fig. 1　AutoPlait "automatically" identifies the dance steps of a motion capture clip, as well as the positions of the all cut points.

べての時系列パターンを表現する手法として，AutoPlait を提案する[*1]．

本論文で扱う問題は以下のとおりである．

問題：大規模時系列シーケンス $\boldsymbol{X}$ が与えられたとき，$\boldsymbol{X}$ を表現する時系列パターンを抽出する．

より具体的には，(a) $\boldsymbol{X}$ の中のパターンの変化点を発見し，部分シーケンス集合（セグメント）に分割し，(b) それらのセグメントをグループ化し，類似時系列パターン（レジーム）を発見する．さらに重要な点として，これらの処理は (c) 高速かつ，自動で行う．

具体例． 図 1 (a) は，*MoCap* データにおける「チキンダンス（chicken dance）」[*2]の時系列シーケンスデータと，AutoPlait の出力結果例である．このモーションは，4 次元のシーケンスで構成され，それぞれの次元が，左右の腕と足の加速度を表現している．チキンダンスは，図 1 (b) に示すとおり，beaks，wings，tail feathers，claps の 4 つの代表的なステップから構成される．図 1 (a) の下の段は，AutoPlait が自動抽出した 4 つのレジームを示している．提案手法は，ダンスに含まれる 4 つのステップを抽出すると同時に，各ステップの切れ目も正しく発見することができる．ここで最も重要なこととして，AutoPlait は，これらの 4 つのステップに関する事前知識を必要とせず，適切な数のレジームとその位置を自動的に把握することができる．

1.1　自動抽出手法の重要性

時系列データを対象とした研究課題は，数多く存在する．パターン発見 [16], [19]，情報要約 [18]，クラスタリング [13]，セグメンテーション [9], [12], [27] や類似シーケンス探索 [20], [22], [26] 等は重要な課題である．しかし，これらの先行研究は，基本的にすべて，パラメータの設定やチューニングが必要である．たとえば，文献 [12], [27] は，セグメントの個数や，エラーの閾値等，いくつかのユーザ指定のパラメータ入力が必要であり，これらのパラメータが出力結果に大きな影響を与える（6 章を参照）．したがって，理想的にはこれらのパラメータ設定やユーザの介入を必要としない手法が望ましい．

また，さらに重要な問題がビッグデータの解析である．時系列データは様々なシステム，アプリケーションにおいて大量に発生している．大規模なデータを解析するにあたり，ユーザの手を介したパラメータ設定を行うことは，多くの時間的コストが必要となるため，現実的ではない．すなわち，ビックデータの解析において，自動処理技術は必要不可欠な重要な要素である．

1.2　本論文の貢献

AutoPlait は以下の特長がある．

(1) 時系列パターン（レジーム）の個数と種類を把握し，それぞれの適切な変化点を発見する．さらに AutoPlait は，パターン変化点の最適解の検出を保証する．

(2) 4 章で述べる提案モデル（MLCM）により，図 1 にあげられるダンスのステップのような，ユーザの直感に合致した時系列パターンの抽出を行う．

(3) AutoPlait はパラメータ設定を必要としない．ユーザの介入を必要とせず，適切なレジームの数，変化点の数を，自動的に発見することができる．

(4) 計算コストは入力データの長さに対して線形である．

2.　関連研究

時系列データの解析に関する研究は様々な分野で進められている [2], [4], [15], [19], [23]．自己回帰モデル（AR: autoregressive model），線形動的システム（LDS: linear dynamical systems），カルマンフィルタ（KF: Kalman filters）は代表的な技術であり，これらに基づく時系列の解析と予測手法が数多く提案されている [8], [13], [24]．Li らは文献 [12] において，欠損を含む大規模時系列シーケンス集合のためのアルゴリズムである DynaMMo を提案している．DynaMMo は LDS に基づき，時系列データのパターンを発見し，シーケンスのセグメント化の能力を持つ．著者らの先行研究 [14] では，主に Web アクセス履歴を用いて，大規模時系列データ集合のための予測手法を提案している．Rakthanmanon らは文献 [20] において，兆単位（"trillions"）の時系列シーケンスを対象とした DTW の類似探索問題を扱っている．

隠れマルコフモデル（HMM: Hidden Markov model）は音声認識を含む様々な分野において，時系列処理手法とし

[*1] ソースコード：
http://www.cs.kumamoto-u.ac.jp/~yasuko/software.html
[*2] Chicken dance:
http://www.youtube.com/watch?v=6UV3kRV46Zs&t=49s

て広く利用されている [28]．HMM に基づく大規模時系列シーケンスのための研究として，文献 [11] では，RFID センサから生成された時系列のマルコフストリームを対象としたイベント問合せの手法が提案され，一方，文献 [7] では大規模 HMM データ集合のための高速探索アルゴリズムが扱われている．最新の研究として，Wang ら [27] は文献 [9] を改良し，pHMM（pattern-based hidden Markov model）を提案している．pHMM は時系列のセグメント化とクラスタリングのための動的モデルであり，時系列シーケンスをマルコフモデルに基づいて線形のセグメントに分割する能力を持つ．階層的確率モデルとして，Fine ら [5] は階層的 HMM（HHMM: hierarchical HMM）を提案し，Fox ら [6] はベータ過程に基づくモデルとして，BP-AR-HMM（beta process autoregressive HMM）を提案している．これらの手法は，時系列の複雑な動的パターンを表現する能力があるが，その一方で，高度なパラメータチューニングや，モデルの構造の定義等が必要となり，さらに，これらの手法は大規模時系列データの解析を想定していない．

情報抽出とクラスタリングの手法は CLARANS [17], BIRCH [29]，TRACLUS [10] を含め，様々なものが提案されている．パラメータフリーな情報解析手法としては，OCI [1] がある．OCI は，外れ値を含む点集合のクラスタリングのための手法である．さらに，文献 [3], [25] においては，MDL の概念を用いて情報要約とクラスタリング問題を扱っている．

3. 問題設定

ここでは本論文で必要な概念について定義を行う．また**表 1** に主な記号と定義を示す．$\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ を d 次元の時系列シーケンスとし，$\boldsymbol{x}_t$ を時刻 t における d 次元ベクトルとする．シーケンス $\boldsymbol{X}$ が与えられたとき，本研究は $\boldsymbol{X}$ を m 個のセグメント集合 $\mathcal{S} = \{s_1, \ldots, s_m\}$ に分割することを目的とする．s_i は i 番目のセグメントの開始点，終了点で構成され（つまり，$s_i = \{t_s, t_e\}$），各セグメントは重複がないものとする．本研究ではさらに，発見したセグメント集合を類似セグメントのグループ（レジーム：regime）に分類する．

定義 1（レジーム） r を最適なセグメントグループの個数とする．それぞれのセグメント s はセグメントグループの 1 つに割り当てられる．これらグループをレジーム（regime）と呼び，それぞれのレジームは統計モデル $\boldsymbol{\theta}_i$ $(i = 1, \ldots, r)$ として表現される．

たとえば，図 1 において，モーションは $m = 8$ 個のセグメントから構成され，それぞれのセグメントが $r = 4$ 個のレジーム（beaks，winds，tail feathers，claps）のうちの 1 つに割り当てられる．

定義 2（セグメントメンバシップ） $\mathcal{F} = \{f_1, \ldots, f_m\}$ を，m 個の整数列とし，f_i を i 番目のセグメントが所属するレジームの番号とする（$1 \le f_i \le r$）．

図 1 では，1 番目のセグメントは 4 番目のレジーム（beaks）に，2 番目のセグメントは 3 番目のレジーム（wings）にそれぞれ所属する．つまり，この場合のセグメントメンバシップは $\mathcal{F} = \{4, 3, 2, 1, 4, 3, 2, 1\}$ となる．

表 1　主な記号と定義
Table 1　Symbols and definitions.

記号	定義
シーケンス	
n	時系列の長さ
d	時系列の次元数
$\boldsymbol{X}$	d 次元の時系列シーケンス
セグメント	
m	$\boldsymbol{X}$ に含まれるセグメントの総数
$\mathcal{S}$	$\boldsymbol{X}$ に含まれるセグメント集合：$\mathcal{S} = \{s_1, \ldots, s_m\}$
$\mathcal{F}$	セグメントメンバシップ：$\mathcal{F} = \{f_1, \ldots, f_m\}$
レジーム	
r	$\boldsymbol{X}$ に含まれるレジームの総数
$\boldsymbol{\Theta}$	r 個のレジームのモデルパラメータ集合：$\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_r, \boldsymbol{\Delta}_{r\times r}\}$
$\boldsymbol{\theta}_i$	i 番目の レジームのモデルパラメータ
k_i	$\boldsymbol{\theta}_i$ の状態数
$\boldsymbol{\Delta}_{r\times r}$	レジーム遷移行列：$\boldsymbol{\Delta} = \{\delta_{ij}\}_{i,j=1}^{r}$
コスト関数	
$\mathcal{C}$	候補解：$\mathcal{C} = \{m, r, \mathcal{S}, \boldsymbol{\Theta}, \mathcal{F}\}$
$Cost_M(\boldsymbol{\Theta})$	$\boldsymbol{\Theta}$ のモデル表現コスト
$Cost_C(\boldsymbol{X}\|\boldsymbol{\Theta})$	$\boldsymbol{\Theta}$ による $\boldsymbol{X}$ の符号化コスト
$Cost_T(\boldsymbol{X};\mathcal{C})$	$\mathcal{C}$ による $\boldsymbol{X}$ の総コスト

本研究の目的は，大規模時系列シーケンスが与えられたときに，そのシーケンスのセグメント化と分割位置の検出および，レジームの発見を高速かつ自動で行うことである．本論文で取り組む問題を以下のように定義する．

問題 1　多次元時系列シーケンス $\boldsymbol{X}$ が与えられたとき，$\boldsymbol{X}$ を表現するような以下の情報を抽出する．

(1) セグメントの総数 m と各セグメントの位置：
$\mathcal{S} = \{s_1, \ldots, s_m\}$
(2) レジームの総数 r とセグメントメンバシップ：
$\mathcal{F} = \{f_1, \ldots, f_m\}$
(3) r 個のレジームを表現するモデルのパラメータ集合：
$\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_r, \boldsymbol{\Delta}_{r\times r}\}$

ここで，これらの情報はコスト関数（式 (5)）を最小化するものを選ぶ．

本論文では，レジームを表現するモデルパラメータ集合 $\boldsymbol{\Theta}$ を，r 個の隠れマルコフモデル（HMM: hidden Markov model），$\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_r\}$，として表現する[*3]．ここでさらに，新たな概念として，**レジーム遷移行列** $\boldsymbol{\Delta}_{r\times r}$ を導入する．レジーム遷移行列（定義 4）とコスト関数（式 (5)）についての詳細は，4 章において示す．

[*3] 本論文で提案する枠組みは，HMM 以外の時系列モデルに適用することも可能である．

問題 1 で示したとおり，本論文の目的は，$\boldsymbol{X}$ の特徴を抽出し，すべての時系列パターンを表現するパラメータ集合 $\{m, r, \mathcal{S}, \boldsymbol{\Theta}, \mathcal{F}\}$ を発見することである．ここで，この全パラメータ集合を**候補解** $\mathcal{C}$ と呼ぶ．

定義 3　$\boldsymbol{X}$ を表現する全パラメータ集合 $\mathcal{C} = \{m, r, \mathcal{S}, \boldsymbol{\Theta}, \mathcal{F}\}$ を候補解と呼ぶ．候補解 $\mathcal{C}$ は，セグメント集合，各セグメントのレジームへの割当て，レジームを表現する確率モデル，これらすべてを表現する．

結論として，本論文の目的は最適な解 $\mathcal{C}$ を発見することである．ここで非常に重要な課題は，(a) どのようにセグメントおよびレジームの数を推定するか，(b) どのようにレジームを表現し，セグメントの割当てを行うかである．本研究では，ユーザによるパラメータ設定を介せず，自動処理によって最適解を求めるための新手法を提案する．

4. データ圧縮と情報要約

本章では，問題 1 を解決するためのモデルを提案する．提案モデルは以下の 2 つのアイデアに基づく．

(1) **多階層連鎖モデル（MLCM: multi-level chain model）**：複数のレジーム間の時系列パターンとその遷移を表現するために，多層的な連鎖モデル（MLCM）を提案する．**図 2** は提案モデルの概念図である．青いセル（state 1,2,3）はレジーム 1 に所属する隠れマルコフモデルの状態遷移を表現し，赤いセルはレジーム 2 に該当する．各レジーム内部においてそれぞれ遷移行列 $\mathbf{A}$ を持つ．たとえばレジームはそれぞれ図 1 のダンスにおける beaks と wings のステップに相当する．ここで，提案モデルの重要な要素として，それぞれのレジーム間のレジーム遷移確率（$\boldsymbol{\Delta}_{r\times r}$）の概念を導入する．レジーム遷移確率は $r \times r$ の行列として表現され，r 個のレジーム間の遷移を表現する（図 2 の例では，$r = 2$ となる）．

(2) **モデル表現コスト**：セグメントとレジームの発見のために，最小記述長（MDL: minimum description length）の概念を用いる．MDL は情報理論に基づくモデル選択基準の 1 つであり，可逆圧縮を行うことができるが，そのものの概念だけでは本論文の目的を直接解決することはできない．そこで，与えられたシーケンス $\boldsymbol{X}$ を適切に表現するモデルを見つけるために，新しい符号体系を定義する．具体的には，(a) 新たな関数（式 (5)）を用いて候補解 $\mathcal{C}$ のモデルコストを推定し，(b) 最適解を発見するための効果的なアルゴリズムを提案する．

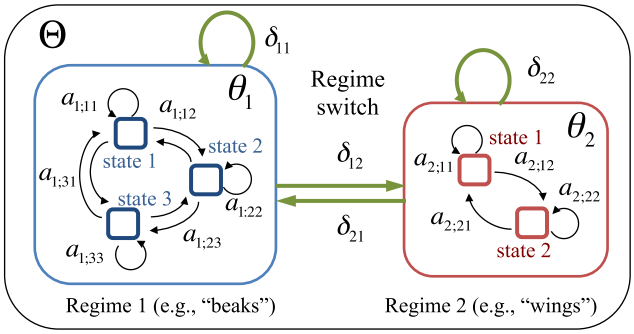


図 2　多階層連鎖モデル $\boldsymbol{\Theta}$（ここでは $r = 2$）

Fig. 2　Multi-level transition diagram for the $r = 2$ regime transition model $\boldsymbol{\Theta}$.

4.1 MLCM：多階層連鎖モデル

多階層連鎖モデル（MLCM: multi-level chain model）は，図 2 にあるように，隠れマルコフの状態遷移をレジームにグループ化し，階層的な時系列パターンの遷移を表現する．本論文では，これ以降，主に 2 層の遷移について焦点を当てるが，2 層以上の多層遷移を表現することも可能である．ここで，図 2 を用いて MLCM の説明を行う．図は，総計 5 つの状態（state）から構成される連鎖モデルであるが，ここでは従来の HMM のように，5×5 の遷移行列を用いるのではなく，上位層の状態（super-state）の概念を導入することによって，パターンのグループ化を行う．ここで，このグループを「レジーム」と呼ぶ．たとえば，図 2 では，3 つの青い状態を用いて beaks のステップを表現し，残りの 2 つの赤い状態を用いて wings のステップを表現する．ここで，各レジームは内部に遷移行列（$a_{1;ji} \in \mathbf{A}_1$，$a_{2;ji} \in \mathbf{A}_2$）を保持し，それらは黒い矢印で表される．さらに，上位層ではレジーム間の遷移行列（$\delta_{vu} \in \boldsymbol{\Delta}_{2\times 2}$）を保有し，これらは緑の矢印で示されている．

定義 4（レジーム遷移行列）　$\boldsymbol{\Delta}_{r\times r}$ を r 個のレジーム群の遷移行列と呼ぶ．ここで，要素 $\delta_{ij} \in \boldsymbol{\Delta}$ は i 番目のレジームから j 番目のレジームへの遷移確率を示す．

行列 $\boldsymbol{\Delta}$ 内部の要素 $\delta_{i,j}$ は確率を表し，$0 \le \delta_{ij} \le 1$，$\sum_j \delta_{ij} = 1$ という条件を持つ．そして提案モデルは r 個のレジーム集合 $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_r, \boldsymbol{\Delta}_{r\times r}\}$ で表現され，$\boldsymbol{\theta}_i$ は i 番目のレジームのモデルパラメータを表現する．ここで，$\boldsymbol{\theta}_i$ は HMM に基づき，初期確率，遷移確率，出力確率の 3 つ組で次のように表現される：$\boldsymbol{\theta}_i = \{\boldsymbol{\pi}_i, \mathbf{A}_i, \mathbf{B}_i\}$ [*4]．

4.2 特徴抽出とデータ圧縮

次に，大規模時系列データを表現するための符号化スキームを導入する．直感的には，データが与えられたときのモデルのよさは次の式で表現できる：$Cost_T = Cost(\mathcal{M}) + Cost(\boldsymbol{X}|\mathcal{M})$．ここで，$Cost(\mathcal{M})$ はモデル $\mathcal{M}$ を表現するためのコストを示し，$Cost(\boldsymbol{X}|\mathcal{M})$ は，$\mathcal{M}$ が与えられたときのデータ $\boldsymbol{X}$ の符号化のコストを示す．

4.2.1 モデル表現コスト

提案モデルの表現コストは以下の要素から構成される．

- 多次元シーケンスデータの長さ n と次元数 d：$\log^*(n) +$

[*4] 本論文では出力確率 $\mathbf{B}$ に多次元ガウス分布を仮定する．これにより多次元ベクトルのシーケンスを確率モデルで表現する（つまり $\mathbf{B} = \{\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)\}_{i=1}^k$）．

$\log^*(d)$ ビット[*5]

- セグメントとレジームの個数 m, r：$\log^*(m)+\log^*(r)$
- 各セグメントのレジームへの割当て（セグメントメンバシップ）：$m\log(r)$ ビット
- 各セグメントの長さ s：$\sum_{i=1}^{m-1}\log^*|s_i|$ ビット
- r 個のレジームのモデルパラメータ集合：$Cost_M(\mathbf{\Theta})$

$$Cost_M(\mathbf{\Theta}) = \sum_{i=1}^{r} Cost_M(\boldsymbol{\theta}_i) + Cost_M(\mathbf{\Delta}). \quad (1)$$

単一のレジームのモデル $\boldsymbol{\theta}$ は，状態数 k（$\log^*(k)$）と確率モデル（$\boldsymbol{\theta}=\{\boldsymbol{\pi},\mathbf{A},\mathbf{B}\}$）の表現コストが必要となる．まとめると，

$$Cost_M(\boldsymbol{\theta}) = \log^*(k) + c_F\cdot(k+k^2+2kd). \quad (2)$$

ここで，c_F は浮動小数点のコストを示す[*6]．同様にして，レジーム遷移行列には，$Cost_M(\mathbf{\Delta}) = c_F\cdot r^2$ のコストを要する．

4.2.2 時系列シーケンスの符号化コスト

先述のとおり，本論文では MLCM モデルを用いてシーケンス $\boldsymbol{X}$ の時系列パターンを表現するが，ここで重要なのは，推定したモデルが $\boldsymbol{X}$ を正しく表現しているかを判断する指標の導入である．ハフマン符号を用いた情報圧縮では，モデル $\boldsymbol{\theta}$ が与えられた際の $\boldsymbol{X}$ の符号化コストを負の対数尤度を用いて次のように表現することができる．

$$Cost_C(\boldsymbol{X}|\boldsymbol{\theta}) = \log_2\frac{1}{P(\boldsymbol{X}|\boldsymbol{\theta})} = -\ln P(\boldsymbol{X}|\boldsymbol{\theta}). \quad (3)$$

ここで，$P(\boldsymbol{X}|\boldsymbol{\theta})$ は $\boldsymbol{X}$ の尤度を示す．シーケンス $\boldsymbol{X}$ と r 個のレジームのモデルパラメータ集合 $\mathbf{\Theta}$ が与えられたとき，データ圧縮のためのコストの総数は次のとおりである．

$$\begin{aligned} Cost_C(\boldsymbol{X}|\mathbf{\Theta}) &= \sum_{i=1}^{m} Cost_C(\boldsymbol{X}[s_i]|\mathbf{\Theta}) \\ &= \sum_{i=1}^{m} -\ln(\delta_{vu}\cdot(\delta_{uu})^{|s_i|-1}\cdot P(\boldsymbol{X}[s_i]|\boldsymbol{\theta}_u)), \end{aligned} \quad (4)$$

ここで，i と $(i-1)$ 番目のセグメントはそれぞれ u と v 番目のレジームに所属し，$f_i=u$，$f_{i-1}=v$，$f_0=f_1$ とする．また，$\boldsymbol{X}[s_i]$ はセグメント s_i の部分シーケンスを示し，$P(\boldsymbol{X}[s_i]|\boldsymbol{\theta}_u)$ はセグメント s_i の尤度とし，$\boldsymbol{\theta}_u$ はセグメント s_i が所属するレジームである．

4.2.3 符号化コスト関数

まとめると，候補解 $\mathcal{C}=\{m,r,\mathcal{S},\mathbf{\Theta},\mathcal{F}\}$ が与えられたときの $\boldsymbol{X}$ の符号長は次のように表現される．

$$\begin{aligned} &Cost_T(\boldsymbol{X};\mathcal{C}) = Cost_T(\boldsymbol{X};m,r,\mathcal{S},\mathbf{\Theta},\mathcal{F}) \\ &= \log^*(n)+\log^*(d)+\log^*(m)+\log^*(r)+m\log(r) \\ &\quad + \sum_{i=1}^{m-1}\log^*|s_i| + Cost_M(\mathbf{\Theta}) + Cost_C(\boldsymbol{X}|\mathbf{\Theta}) \end{aligned} \quad (5)$$

したがって本論文の次の目標は，上記のコスト関数を最小化するようなセグメントおよびレジーム集合を発見することであり，次章ではそのためのアルゴリズムについて述べる．

5. 最適化アルゴリズム

前章では，候補解 $\mathcal{C}=\{m,r,\mathcal{S},\mathbf{\Theta},\mathcal{F}\}$ が与えられたうえでシーケンス $\boldsymbol{X}$ を表現するためのコスト関数として，式 (5) を示した．続いて本章では，式 (5) に基づき，最適な解 $\mathcal{C}$ を発見するためのアルゴリズムを提案する．

5.1 概要

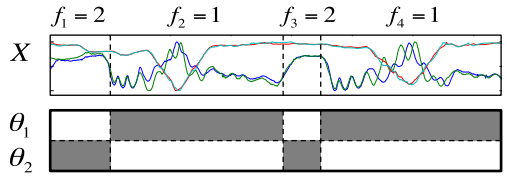
本研究では，前章で述べたコストモデルに基づき，セグメントおよびレジームの個数を自動的に選択する．直感的には，データの圧縮率が高ければ，そのモデルはデータに含まれるパターンをよく表現しているといえる．つまり，セグメントの個数 m，レジームの個数 r，モデルパラメータ集合 $\mathbf{\Theta}$，そしてメンバシップ $\mathcal{F}$ から構成される候補解 $\mathcal{C}$ に対し，$\boldsymbol{X}$ の符号化コスト $Cost_T(\boldsymbol{X};m,r,\mathcal{S},\mathbf{\Theta},\mathcal{F})$ が最小となるとき，$\mathcal{C}$ は最適なモデルになる．

次に，具体的な最適化手法を示す．ここでは問題を簡略化するため，次にあげる 3 つの部分問題に分割する．(1) レジームの個数を $r=2$ に固定し，各レジームのモデルパラメータも与えられる場合を考える．(2) $r=2$ を固定した状態で，モデルパラメータの推定を行う．(3) 最適なレジームの個数を推定する．より具体的には各部分問題に対応し，以下の 3 つのアルゴリズムを提案する．

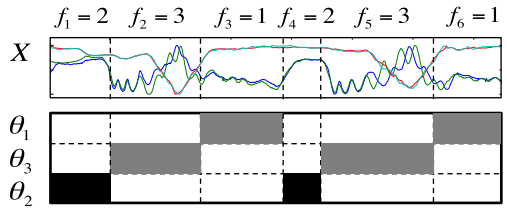
(1) **CutPointSearch**：レジームの個数（$r=2$）とモデルパラメータが与えられたときに，$\boldsymbol{X}$ を 2 つのレジームに分割し，それぞれのセグメントの分割位置を検出する．
(2) **RegimeSplit**：レジームの個数 $r=2$ が与えられたときに，2 つのレジームを表現するモデルパラメータ（$\boldsymbol{\theta}_1$，$\boldsymbol{\theta}_2$，$\mathbf{\Delta}$）を推定する．
(3) **AutoPlait**：最適なレジームの個数（$r=2,3,4,\ldots$）を求める．

図 3 は，AutoPlait の処理のながれである．AutoPlait は $r=2$ から開始し，セグメントとレジームを分割しながらシーケンス $\boldsymbol{X}$ を適切に表現する解 $\mathcal{C}$ を発見する．レジームとセグメントを分割するとき，コスト関数（図 3 (d)）が下がる．例外として，図中における赤丸の箇所（Iteration 3）については，レジーム $\boldsymbol{\theta}_1$ が分割しないた

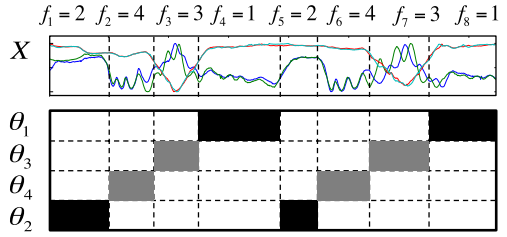
[*5] ここで，$\log^*$ は整数のユニバーサル符号長を表す：$\log^*(x)\approx\log_2(x)+\log_2\log_2(x)+\ldots$ [21].
[*6] 本論文では 4×8 ビットとする．



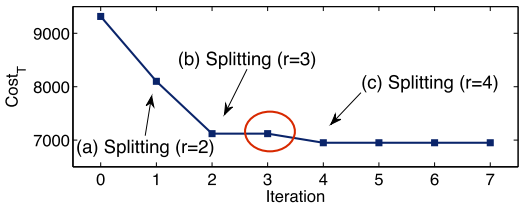
(a) Iteration 1（レジーム：$r=2$，セグメント：$m=4$）



(b) Iteration 2（レジーム：$r=3$，セグメント：$m=6$）



(c) Iteration 4（レジーム：$r=4$，セグメント：$m=8$）



(d) 総コストの変化

図 3　AutoPlait の概要図：AutoPlait は $\boldsymbol{X}$ が与えられたとき，反復処理により適切なセグメント/レジームの個数を求める

Fig. 3　Overview of the workflow of AutoPlait: (a)–(c) starting with sequence $\boldsymbol{X}$, our algorithm iteratively finds the segment groups (i.e., regimes), and their segments.

め，コストが下がらない．

5.2　CutPointSearch

まず最も単純な部分問題として，シーケンス $\boldsymbol{X}$ と，2 つのレジームのモデルパラメータ $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\Delta}\}$ が与えられている場合を考える．CutPointSearch は レジームのモデルパラメータに基づき，$\boldsymbol{X}$ のパターンの変化点（つまりセグメントの分割位置）を検出することができる．ここで重要な点として，提案アルゴリズムは探索漏れがないことを保証しながらも，高速かつ単一の走査によって，最適なレジーム変化点の個数と位置を検出することができる．図 4 はレジームの変化点が 1 つの場合における CutPointSearch の処理の様子を示している．これは青いレジーム $\boldsymbol{\theta}_1$ から赤いレジーム $\boldsymbol{\theta}_2$ へ切り替わる例である．

より具体的には，ここでは $\boldsymbol{X}$ が与えられたとき，コスト関数（式 (4)）を最小とするような，複数のレジーム変化点を発見し，それらを 2 つのセグメント集合（$\mathcal{S}_1$，$\mathcal{S}_2$）に分割することを考える．ここで，2 つのセグメント集合は，偶数番目のセグメントは 1 つ目のレジームに，奇数番目は 2 つ目のレジームに所属する．たとえば，図 3 (a) では $\mathcal{S}_1 = \{s_2, s_4\}$ は $\boldsymbol{\theta}_1$ に，$\mathcal{S}_2 = \{s_1, s_3\}$ は $\boldsymbol{\theta}_2$ にそれぞれ所属する．

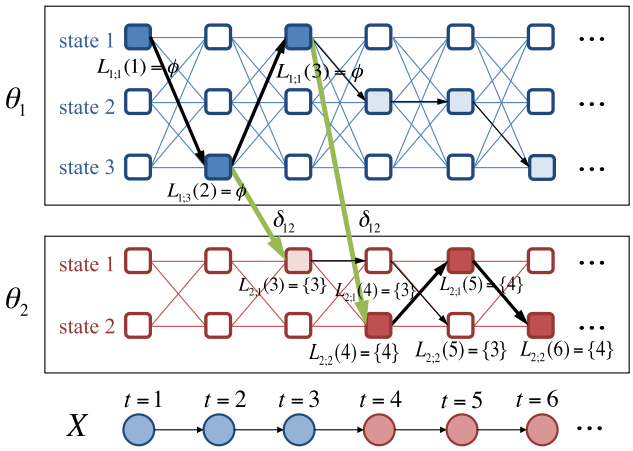


図 4　CutPointSearch の様子．$\boldsymbol{X}$ と $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ が与えられたとき，単一の走査でレジームの変化点（$t=4$）を検出する

Fig. 4　Illustration of CutPointSearch. Given a sequence $\boldsymbol{X}$ and two models $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ (here, duration $n=6$), our algorithm requires only a single scan to detect the regime cut point (i.e., at time-tick $t=4$).

ここで，複数の変化点を検出するために，多階層連鎖モデル（MLCM）の概念を用いる．図 4 のように 2 つのレジーム $\boldsymbol{\theta}_1$，$\boldsymbol{\theta}_2$ が与えられているとする．ここで，この 2 つのレジームは相互に遷移確率を持つ．モデルが与えられたうえでの符号化コスト $Cost_C(\boldsymbol{X}|\boldsymbol{\Theta}) = -\ln P(\boldsymbol{X}|\boldsymbol{\Theta})$ を計算するために，本論文では動的計画法に基づくアルゴリズムを提案する．

5.2.1　アルゴリズム

シーケンス $\boldsymbol{X}$ と 2 つのレジーム $\boldsymbol{\theta}_1 = \{\boldsymbol{\pi}_1, \mathbf{A}_1, \mathbf{B}_1\}$, $\boldsymbol{\theta}_2 = \{\boldsymbol{\pi}_2, \mathbf{A}_2, \mathbf{B}_2\}$，およびレジーム遷移行列 $\boldsymbol{\Delta} = \{\delta_{11}, \delta_{12}, \delta_{21}, \delta_{22}\}$ が与えられたとき，$\boldsymbol{X}$ の尤度 $P(\boldsymbol{X}|\boldsymbol{\Theta})$ は次のように計算される．

$$P(\boldsymbol{X}|\boldsymbol{\Theta}) = \max \begin{cases} \max_{1\le i\le k_1}\{p_{1;i}(n)\} & // \text{ regime } \boldsymbol{\theta}_1 \\ \max_{1\le u\le k_2}\{p_{2;u}(n)\} & // \text{ regime } \boldsymbol{\theta}_2 \end{cases} \tag{6}$$

$$p_{1;i}(t) = \max \begin{cases} \delta_{21}\cdot \max_v\{p_{2;v}(t-1)\}\cdot \pi_{1;i}\cdot b_{1;i}(\boldsymbol{x}_t) \\ \qquad // \text{ regime switch from } \boldsymbol{\theta}_2 \text{ to } \boldsymbol{\theta}_1 \\ \delta_{11}\cdot \max_j\{p_{1;j}(t-1)\cdot a_{1;ji}\}\cdot b_{1;i}(\boldsymbol{x}_t) \\ \qquad // \text{ staying at regime } \boldsymbol{\theta}_1 \end{cases} \tag{7}$$

$$p_{2;u}(t) = \max \begin{cases} \delta_{12} \cdot \max_j\{p_{1;j}(t-1)\} \cdot \pi_{2;u} \cdot b_{2;u}(\boldsymbol{x}_t) \\ \qquad // \text{ regime switch from } \boldsymbol{\theta}_1 \text{ to } \boldsymbol{\theta}_2 \\ \delta_{22} \cdot \max_v\{p_{2;v}(t-1) \cdot a_{2;vu}\} \cdot b_{2;u}(\boldsymbol{x}_t) \\ \qquad // \text{ staying at regime } \boldsymbol{\theta}_2 \end{cases} \tag{8}$$

ここで，$p_{1;i}(t)$ は，時刻 t におけるレジーム $\boldsymbol{\theta}_1$ の状態 i の確率の最大値であり，$p_{2;u}(t)$ は，時刻 t における $\boldsymbol{\theta}_2$ の状態 u の確率の最大値を示す．時刻 $t=1$ においては，各レジームの確率は次のように計算する．

$$\begin{aligned} p_{1;i}(1) &= \delta_{11} \cdot \pi_{1;i} \cdot b_{1;i}(\boldsymbol{x}_1) \\ p_{2;u}(1) &= \delta_{22} \cdot \pi_{2;u} \cdot b_{2;u}(\boldsymbol{x}_1) \end{aligned} \tag{9}$$

式 (7) の上段は，レジーム間の切替え（$\boldsymbol{\theta}_2$ から $\boldsymbol{\theta}_1$）の確率，下段はレジーム $\boldsymbol{\theta}_1$ の内部の状態遷移の確率を示し，さらに，式の内部の各要素は次のとおりである．

- δ_{21}：$\boldsymbol{\theta}_2$ から $\boldsymbol{\theta}_1$ へのレジーム遷移確率
- $\max_v\{p_{2,v}(t-1)\}$：時刻 $t-1$ における $\boldsymbol{\theta}_2$ 内の確率の最大値
- $\pi_{1,i}$：$\boldsymbol{\theta}_1$ 内の状態 i の初期確率
- $b_{1;i}(\boldsymbol{x}_t)$：$\boldsymbol{\theta}_1$ 内の状態 i における $\boldsymbol{x}_t$ の出力確率
- $a_{1;ji}$：$\boldsymbol{\theta}_1$ 内の状態 j から状態 i への遷移確率

同様に，式 (8) はレジーム $\boldsymbol{\theta}_2$ における確率の最大値を示す．

続いて，レジームの変化点の候補集合からどのように最適解を発見するかについて述べる．$\mathcal{L} = \{l_1, l_2, \ldots, l_{m-1}\}$ を変化点の位置情報の集合とする．ここで m はセグメントの個数，l_i は i 番目の変化点（つまり $1 \le l_i \le n$）とする．本論文では，これらの候補集合を 2 つのレジームの各状態に対して保持することで，最適な変化点集合を発見する．

$$\mathcal{L}_{1;i}(t) = \begin{cases} \mathcal{L}_{2;v}(t-1) \cup \{t\} & // \text{ switch from } \boldsymbol{\theta}_2 \text{ to } \boldsymbol{\theta}_1 \\ \mathcal{L}_{1;j}(t-1) & // \text{ staying at regime } \boldsymbol{\theta}_1 \end{cases} \tag{10}$$

$$\mathcal{L}_{2;u}(t) = \begin{cases} \mathcal{L}_{1;j}(t-1) \cup \{t\} & // \text{ switch from } \boldsymbol{\theta}_1 \text{ to } \boldsymbol{\theta}_2 \\ \mathcal{L}_{2;v}(t-1) & // \text{ staying at regime } \boldsymbol{\theta}_2 \end{cases} \tag{11}$$

ここで $\mathcal{L}_{1;i}(t)$ は時刻 t における $\boldsymbol{\theta}_1$ の状態 i の変化点集合であり，$\mathcal{L}_{2;u}(t)$ は $\boldsymbol{\theta}_2$ の状態 u の変化点集合である．これらは式 (7)，(8) の尤度計算に基づき更新される．変化点は，レジームが他方に切り替わった場合，その時刻 t を変化点の候補として集合に加える．アルゴリズム 1 は CutPointSearch の具体的な処理を示す．CutPointSearch は時刻 $t=n$ において $\mathcal{L}_{1;i}(n)$ と $\mathcal{L}_{2;u}(n)$ 内のすべての状態 i，u の中から，尤度 $P(\boldsymbol{X}|\boldsymbol{\Theta})$ を最大化するような最適解 $\mathcal{L}_{best}$ を選出する．

例 1　ここでは図 4 を用いてアルゴリズムの具体的な例を示す．時刻 $t=1$ と $t=2$ において，$\boldsymbol{\theta}_1$ 内の確率 $p_{1;1}(1)$，$p_{1;3}(2)$ はそれぞれ確率の最大値を持つ．時刻 $t=3$ において CutPointSearch は $p_{1;3}(2)$ から $p_{2;1}(3)$ への変化点の候補を発見する（図では $\boldsymbol{\theta}_1$ から $\boldsymbol{\theta}_2$ への緑の矢印）．同時に，$\mathcal{L}_{2;1}(3) = \{3\}$ を候補集合として更新する．同様にして，2 番目の候補点として $\mathcal{L}_{2;2}(4) = \{4\}$ を保持する．時刻 $t=6$ において，アルゴリズムは $p_{2;2}(6)$ が確率の最大値を持つことを明らかにし，$\mathcal{L}_{2;2}(6) = \{4\}$ を最適解として出力する．

Algorithm 1 CutPointSearch $(\boldsymbol{X}, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\Delta})$

```
Input: Sequence X, model parameters of two regimes {θ1, θ2, Δ}
Output: (a) Number of segments assigned to each regime, m1, m2
        (b) Segment sets of two regimes S1, S2
/* Compute p1;i(t) and p2;u(t) */
for t = 1 : n do
    Compute p1;i(t) for state i = 1, ..., k1;    /* Equations 7 and 9 */
    Compute p2;u(t) for state u = 1, ..., k2;    /* Equations 8 and 9 */
    Update L1;i(t) for state i = 1, ..., k1;     /* 10 */
    Update L2;u(t) for state u = 1, ..., k2;     /* 11 */
end for
/* Divide into two sets of segments S1, S2 */
Choose the best cut-point set L_best;
t_s = 1; /* Starting position of first segment */
for each cut point l_i in L_best do
    Create segment s_i = {t_s, l_i};
    if i is odd then
        Add s_i into S1;   m1 = m1 + 1;
    else
        Add s_i into S2;   m2 = m2 + 1;
    end if
    t_s = l_i;
end for
return {m1, m2, S1, S2};
```

5.2.2　理論的な分析

補助定理 1　CutPointSearch は $O(ndk^2)$ の計算量を要する．

証明 1　CutPointSearch は各時刻において $O(dk^2)$ 個の確率を計算する．変化点の検出には単一の走査のみが必要となるため，まとめると計算量は $O(ndk^2)$ となる．

補助定理 2　与えられたモデル $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_r, \boldsymbol{\Delta}\}$ に対し，CutPointSearch は最適な変化点集合を検出することを保証する．

証明 2　まず，k 個の状態を持つレジームが r 個ある場合を考える．ここで，各状態は r 個のレジームに存在するすべての $k \times r$ の状態と接続していると想定する．さらに，$\delta'_{u,i;v,j}$ はレジーム u の状態 i からレジーム v の状態 j への遷移確率を示すとする．これは，単一のレジームにおいて $k \times r$ 個の状態を保持することを同じである．コストを

Algorithm 2 RegimeSplit ($\boldsymbol{X}$)

Input: Sequence $\boldsymbol{X}$
Output: (a) Number of segments assigned to each regime, m_1, m_2
(b) Segment sets of two regimes, $\mathcal{S}_1, \mathcal{S}_2$
(c) Model parameters of two regimes $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\Delta}\}$
Initialize models $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$;　/* Equation 12 */
while improving the cost **do**
　/* Find segments (phase 1) */
　$\{m_1, m_2, \mathcal{S}_1, \mathcal{S}_2\}$ = CutPointSearch ($\boldsymbol{X}, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\Delta}$);
　/* Update model parameters (phase 2) */
　$\boldsymbol{\theta}_1$ = BaumWelch ($\boldsymbol{X}[\mathcal{S}_1]$);
　$\boldsymbol{\theta}_2$ = BaumWelch ($\boldsymbol{X}[\mathcal{S}_2]$);
　Update regime transitions $\boldsymbol{\Delta}$;　/* Equation 13 */
end while
return $\{m_1, m_2, \mathcal{S}_1, \mathcal{S}_2, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\Delta}\}$;

最小化するような最適なパスは，動的計画法を用いて見つけることができる．ここで，提案モデル（MLCM）において，レジーム u はレジーム v に接続しているため，つまり，状態 i と j のすべての組合せにおける遷移確率は $\delta_{uv} = \delta'_{u,i;v,j}$ として表現することができる．したがって，CutPointSearch は最適なパスとコストを最小化する最適な変化点集合を検出することができる．

5.3 RegimeSplit

これまでは 2 つのレジームのモデルパラメータ $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\Delta}\}$ が与えられている場合について考えた．次に考える問題は，モデルパラメータの推定である．具体的には，(a) 2 つのレジームのモデルパラメータを推定し，同時に，(b) すべてのレジーム変化点を検出したい．そこで本研究では，式 (5) を用いてシーケンス $\boldsymbol{X}$ の表現コストを最小にするようなモデルパラメータの推定を行う．アルゴリズム 2 は RegimeSplit の処理を示す．提案アルゴリズムは以下に示す 2 つのステップから構成される反復処理によって，モデルパラメータの推定を行う．

- ステップ *1*：CutPointSearch （アルゴリズム 1）を利用し，符号化コストが最小となるレジーム変化点を検出し，セグメント集合を 2 つのグループ $\{\mathcal{S}_1, \mathcal{S}_2\}$ に分割する．
- ステップ *2*：ステップ 1 で得られたセグメント集合に基づき，2 つのレジームのモデルパラメータ $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\Delta}\}$ を推定する．ここで，HMM のパラメータの学習には，Baum-Welch アルゴリズムを用いる．

モデルパラメータの初期化． RegimeSplit では，はじめにモデルパラメータ $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$ を初期化する必要がある．最も簡易な方法としては，シーケンス $\boldsymbol{X}$ の中に含まれる部分シーケンスをランダム抽出し，モデルの初期値に設定することである．しかし，この方法を用いる場合，初期値に大きく依存するため局所解へ収束してしまう可能性がある．そこで，本研究ではこの問題を解決するため，サンプリングに基づく手法を提案する．まず $\boldsymbol{X}$ の中から複数個のセグメント/部分シーケンスをサンプルとして均等に取り出す．次に，それぞれのサンプルセグメント s に対し，モデルパラメータ $\boldsymbol{\theta}_s$ を推定する．続いて，すべてのモデルのペア $\{\boldsymbol{\theta}_{s_1}, \boldsymbol{\theta}_{s_2}\}$ に対し，符号化コストを計算し，最も適切なペア $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$ を初期モデルとして選出する．

$$\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\} = \underset{\boldsymbol{\theta}_{s_1}, \boldsymbol{\theta}_{s_2} | s_1, s_2 \in \mathcal{X}}{\arg\min} \; Cost_C(\boldsymbol{X} | \boldsymbol{\theta}_{s_1}, \boldsymbol{\theta}_{s_2}), \qquad (12)$$

ここで，$\mathcal{X} = \{s_1, s_2, \ldots\}$ は，$\boldsymbol{X}$ から取り出したサンプルの集合を示す．

モデルパラメータの推定． HMM のモデルパラメータの推定手法である Baum-Welch アルゴリズムは，モデル $\boldsymbol{\theta}$ に対し，隠れ状態の数 k を与える必要がある．しかし，この k を手動で設定するのは非常に難しい．もし k の値を小さくすれば，データの表現能力が低くなり，適切なセグメントおよびレジームを求めることが困難となる．一方で，もし k を大幅に上げてしまうと，オーバフィッティングを招く．そこで本研究では，隠れ状態の個数を $k = 1, 2, 3, \ldots$ のように変化させながら，コスト関数 $Cost_M(\boldsymbol{\theta}) + Cost_C(\boldsymbol{X}[\mathcal{S}] | \boldsymbol{\theta})$ が最小となるような k を求める．

さらに，本研究ではレジーム遷移確率 $\boldsymbol{\Delta}$ についても符号化コストを最小にする必要がある．そこで，セグメントの変化点集合 $\{\mathcal{S}_1, \mathcal{S}_2\}$ とモデルパラメータ $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$ が与えられたときのレジーム遷移確率 $\boldsymbol{\Delta} = \{\delta_{11}, \delta_{12}, \delta_{21}, \delta_{22}\}$ をラグランジュ乗数法に基づき次のように計算する．

$$\delta_{11} = \frac{\sum_{s \in \mathcal{S}_1} |s| - N_{12}}{\sum_{s \in \mathcal{S}_1} |s|}, \qquad \delta_{12} = \frac{N_{12}}{\sum_{s \in \mathcal{S}_1} |s|}, \qquad (13)$$

ここで $\sum_{s \in \mathcal{S}_1} |s|$ はレジーム $\boldsymbol{\theta}_1$ に所属するセグメントの長さの総和を示し，N_{12} は $\boldsymbol{\theta}_1$ から $\boldsymbol{\theta}_2$ へのレジームの切替え回数を示す．δ_{21}，δ_{22} についても同様に計算できる．

5.4 AutoPlait

本論文の最終目標は，問題 1 で述べたとおり，大規模時系列データの中から任意の数のパターンを自動的に抽出することである．ここで解決すべき問題は，(a) 最適なセグメントおよびレジームの個数 m，r はどのように決定すればよいか，(b) それぞれのセグメントを適切なレジームに割り当てるにはどうしたらいいか，の 2 点である．

5.4.1 アルゴリズム

大規模時系列シーケンスの中から自動的にパターンを取り出す手法として AutoPlait を提案する．AutoPlait はスタックを用いた手法であり．貪欲法に基づくアルゴリズムである．AutoPlait は与えられたシーケンスをセグメントに分割し，新たなレジームを生成し，コスト関数である式 (5) を減少させていく．アルゴリズム 3 は AutoPlait の処理の流れを示している．各ステップにおいて，

Algorithm 3 AutoPlait ($\boldsymbol{X}$)

Input: Sequence $\boldsymbol{X}$
Output: Complete set of parameters $\mathcal{C}$, i.e.,
　　(a) Number of segments, m
　　(b) Number of regimes, r
　　(c) Segment set $\mathcal{S} = \{s_1, \ldots, s_m\}$
　　(d) Model parameters of regimes $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_r; \boldsymbol{\Delta}\}$
　　(e) Segment membership $\mathcal{F} = \{f_1, \ldots, f_m\}$
$\mathcal{Q} = \emptyset$;　/* $\mathcal{Q}$: stack for number of segments, segment set, regime */
$\mathcal{S} = \emptyset$;　$m = 0$;　$r = 0$;　$\mathcal{S}_0 = \{1, n\}$;　$m_0 = 1$;
$\boldsymbol{\theta}_0$ = BaumWelch ($\boldsymbol{X}[\mathcal{S}_0]$);　/* Estimate model $\boldsymbol{\theta}_0$ of $\mathcal{S}_0$ */
Push an entry $\{m_0, \mathcal{S}_0, \boldsymbol{\theta}_0\}$ into $\mathcal{Q}$;
while stack $\mathcal{Q} \neq \emptyset$ **do**
　Pop an entry $\{\boldsymbol{\theta}_0, m_0, \mathcal{S}_0\}$ from $\mathcal{Q}$;
　/* Try to refine a regime */
　$\{m_1, m_2, \mathcal{S}_1, \mathcal{S}_2, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\Delta}\}$ = RegimeSplit ($\boldsymbol{X}[\mathcal{S}_0]$);
　/* Compare single regime $\boldsymbol{\theta}_0$ v.s. regime pair $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ */
　if $Cost_T(\boldsymbol{X}; \mathcal{S}_0, \boldsymbol{\theta}_0) > Cost_T(\boldsymbol{X}; \mathcal{S}_1, \mathcal{S}_2, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$ **then**
　　/* Regime pair win - split regime */
　　Push entries $\{m_1, \mathcal{S}_1, \boldsymbol{\theta}_1\}$, $\{m_2, \mathcal{S}_2, \boldsymbol{\theta}_2\}$ into $\mathcal{Q}$;
　else
　　/* Single regime win - no more split, leave it out of the stack */
　　$\mathcal{S} = \mathcal{S} \cup \mathcal{S}_0$;　$\boldsymbol{\Theta} = \boldsymbol{\Theta} \cup \boldsymbol{\theta}_0$;　$r = r + 1$;
　　Update $\boldsymbol{\Delta}_{r \times r}$;　/* 13 */
　　$f_i = r$　$(i = m + 1, \ldots, m_0)$;　$m = m + m_0$;
　end if
end while
return $\mathcal{C} = \{m, r, \mathcal{S}, \boldsymbol{\Theta}, \mathcal{F}\}$;

AutoPlait はスタック $\mathcal{Q}$ の中からエントリ $\{\boldsymbol{\theta}_0, m_0, \mathcal{S}_0\}$ を取り出す．続いて，現在のレジーム $\boldsymbol{\theta}_0$ の分割を試み，新たなレジームの候補ペア $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$ とそのセグメント集合 $\{\mathcal{S}_1, \mathcal{S}_2\}$ を生成する．もし新しいレジームの候補のコストが現在のレジームのコストより低い場合は（つまり，レジームの候補ペアが勝った場合），AutoPlait は候補ペアをスタック $\mathcal{Q}$ に追加する．もし現在のレジームのコストのほうが低ければ，エントリをスタックから取り除き，$\{\boldsymbol{\theta}_0, m_0, \mathcal{S}_0\}$ を出力する．これらの処理をスタックが空になるまで繰り返す．

5.4.2 理論的な分析

補助定理 3　AutoPlait の計算量はシーケンスの長さ n に対し線形である．

証明 3　各反復処理において，CutPointSearch と RegimeSplit は符号化コストとモデルパラメータの推定のために $O(ndk^2)$ の計算量を要する．ここで，d は次元数，k はレジーム $\{\boldsymbol{\theta}_i\}_{i=1}^{r}$ の中の隠れ状態の数の最大値（つまり $k = \max\{k_1, k_2, \ldots, k_r\}$）を示す．よって，AutoPlait の計算量は $O(\#iter \cdot ndk^2)$ である．ここで，反復回数 $\#iter$，隠れ状態の個数 k と次元数 d は非常に小さい定数であるため，無視することができる．よって，計算量は $O(n)$ である．

6. 評価実験

本論文では AutoPlait の有効性を検証するため，実データを用いた実験を行った．具体的には，本章では以下の項目について検証する．

Q1　時系列パターン抽出に関する提案手法の有効性
Q2　レジーム抽出と変化点検出に対する精度の検証
Q3　パターン抽出に対する計算時間の検証

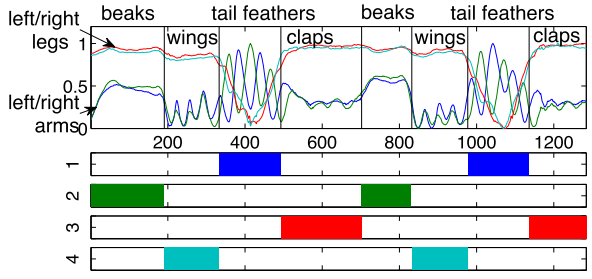
実験は 32 GB のメモリ，Intel Core 2 Duo 1.86 GHz の CPU を搭載した Linux のマシン上で実施した．本論文では以下の 3 つの実データを用いて検証を行った．各データは平均値と分散値で正規化（z-normalization）して使用した．

- *MoCap*：*MoCap* は，1 秒 120 フレームでヒトの動きを計測したモーションキャプチャのデータセットである[*7]．本実験ではデータの中から左右の腕と足の 4 次元から構成される加速度の値を使用した．
- *WebClick*：このデータセットは，1 カ月間（2007/4/1–4/30）のウェブアクセス履歴である．このデータは URL ID（1,797 URLs），user ID（2,582,252 ユーザ），time の 3 つの属性から構成される．URL には，blog，news をはじめとする様々な種類のウェブサイトが含まれる．
- *GoogleTrend*[*8]：このデータセットは，Google による検索クエリの頻度を週ごとに 9 年間にわたり集計したものである．各シーケンスは各クエリの出現頻度を表す．

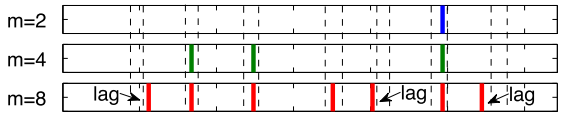
6.1 時系列データからの特徴抽出

ここでは，提案手法である AutoPlait の情報抽出の効果を検証するため，大規模時系列データの解析のための最新の手法として DynaMMo [12]，pHMM [27] と比較した．**図 5** は *MoCap* データにおけるチキンダンスに対する特徴抽出の結果を示している．1 章の図 1 においてもすでに同種のダンスの解析結果を示しているが，ここで使用したデータは，図 5 とは異なるユーザの動きから生成したデータである．AutoPlait は，両データに対し，$m = 8$ 個のセグメント，$r = 4$ 個のステップの抽出に成功している．一方，図 5 (b) は DynaMMo によるセグメントの分割結果である．DynaMMo はユーザの指定するパラメータとして，セグメントの個数 m を与える必要があるため，ここでは $m = 2, 4, 8$ についてそれぞれ結果を示している．図 5 (b) では，点線が正解値を示し，色のついた実線が DynaMMo によって得られた値である．DynaMMo は正しいセグメントの個数として $m = 8$ を与えた場合であっても，変化点の位置を正確に特定することができない．さらに重要な

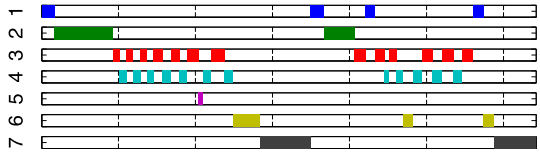
[*7] http://mocap.cs.cmu.edu/
[*8] http://www.google.com/insights/search/



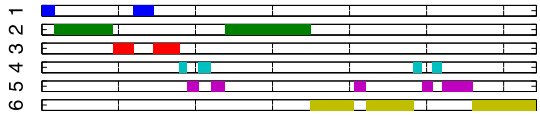
(a) AutoPlait（パラメータ不要）



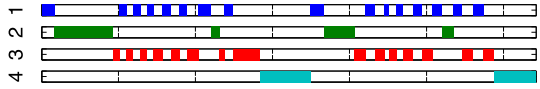
(b) DynaMMo（パラメータ：$m = 2, 4, 8$）



(c-1) pHMM（パラメータ：$\epsilon_r = 0.1$，$\epsilon_c = 0.8$）



(c-2) pHMM（パラメータ：$\epsilon_r = 1.0$，$\epsilon_c = 0.8$）



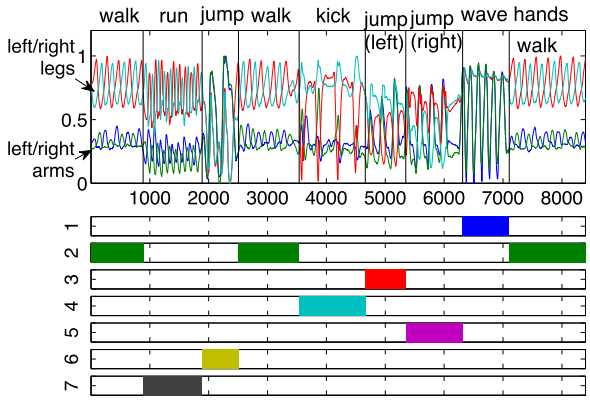
(c-3) pHMM（パラメータ：$\epsilon_r = 0.1$，$\epsilon_c = 4.0$）

図 5　*MoCap* における AutoPlait，DynaMMo，pHMM の出力結果

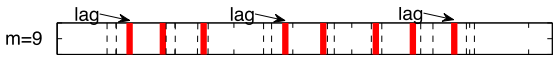
Fig. 5　AutoPlait is fully automatic.

こととして，DynaMMo はクラスタリングの能力がないため，レジームを発見することができない．図 5 (c1)–(c3) は pHMM の出力結果である．pHMM は，モデル学習の際のエラー値に関連する閾値（すなわちパラメータ）として ϵ_r と ϵ_c の 2 つを設定しなくてはならない．たとえば，図 5 (c1) では pHMM は 36 個のセグメントと 7 つのクラスタを発見している．pHMM は線形のパターンをモデル化する手法であるため，複雑な時系列パターンや長期的なトレンドを抽出することができない．図 5 (c1)–(c3) において，pHMM はユーザの設定するパラメータに対し，出力が大きく左右されることが分かる．

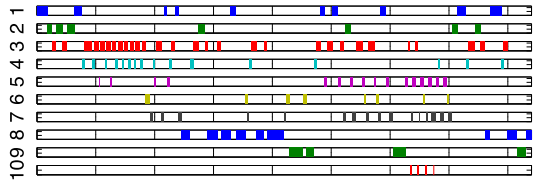
図 6 は，より複雑なパターンに対する特徴抽出の例である．このシーケンスは walking, jumping, kicking 等の複数のパターンを含み，walking 以外のパターンは 1 度しか出現しない．従来のクラスタリング手法にはない AutoPlait の強みの 1 つとして，重複のないトレンドの検出があげられる．実際に，図 6 (a) において AutoPlait は running, jumping のような，繰り返し出現しないモーションを抽出



(a) AutoPlait（パラメータ不要）



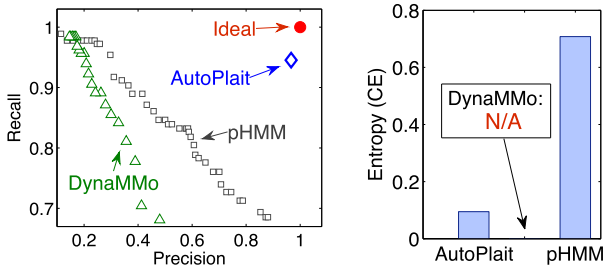
(b) DynaMMo（パラメータ：$m = 9$）



(c) pHMM（パラメータ：$\epsilon_r = 0.1$，$\epsilon_c = 0.8$）

図 6　*MoCap* における AutoPlait の出力結果のようす

Fig. 6　AutoPlait can detect all distinct motions.



(a) 適合率と再現率　　(b) CE スコア

図 7　AutoPlait の精度

Fig. 7　Accuracy of AutoPlait.

している．図 6 (b)，(c) は，DynaMMo と pHMM の出力結果である．図 5 と同様に，これらの手法はレジームの発見に失敗している．

6.2　精度

続いて，与えられたシーケンスに対する提案手法の変化点抽出とクラスタリングの精度について検証する．

変化点抽出の精度． **図 7** (a) は，提案手法と比較手法におけるセグメントの分割点の抽出の精度を適合率と再現率に基づき比較したものである．ここでは *MoCap* データから合計 20 個のシーケンスを選び使用した．各シーケンスにはそれぞれ平均して 10 個程度のセグメントが含まれており，本実験で使用したデータはおよそ $n = 20 \times 10{,}000$ のモーションフレームを含んでいる．適合率は，抽出された

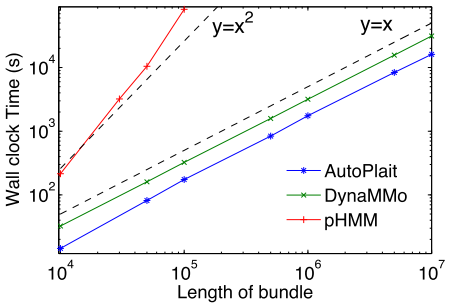


図 8　AutoPlait の計算コスト

Fig. 8　AutoPlait scales linearly: wall clock time vs. datasize n.

変化点の合計数とそのうち正解であった点の合計数の割合を示す．再現率は，すべての変化点の正解値の数と抽出された点の中で正解した合計数の割合を示す．両者とも，精度が高い場合は 1 に近づく．図 7 (a) では，AutoPlait は 1 点のみで表現されている．これは，提案手法がパラメータを持たず，出力結果が 1 つに定まるためである．図のように，提案手法は，95% 以上の正解率を示している．

一方，DynaMMo は，図 5 で示したとおり，ユーザの設定するパラメータ m を要する．ここではパラメータを $m = 2, 4, 6, \ldots, 30$ のように変化させてそれぞれ精度を比較している．同様に，pHMM はモデルの学習精度に関連する閾値のパラメータを要する．本実験では ϵ_r を 0.1 から 10.0 に変化させながら精度を検証した．DynaMMo と pHMM はパラメータによって適合率と再現率が大きく左右されることが分かる．

クラスタリング精度． AutoPlait は与えられたシーケンスの中からレジームを発見すると同時に，各クラスタがどのレジームに所属するかの情報（つまり，セグメントメンバシップ）を抽出することができる．図 7 (b) は提案手法と比較手法におけるクラスタリングの精度を示している．より具体的には，レジームの正解ラベルおよび推定したラベルに関する混合行列（CM: confusion matrix）を作成し，条件付きエントロピー（CE: conditional entropy）のスコアを次のように計算した：$CE = -\sum_{i,j} \frac{CM_{ij}}{\sum_{ij} CM_{ij}} \log \frac{CM_{ij}}{\sum_{j} CM_{ij}}$．正確にラベルを求めることができた場合，混合行列 CM は対角行列となり，$CE = 0$ となる．図 7 (b) は，AutoPlait と pHMM における CE の平均スコアを示している．pHMM と異なり，提案手法はほぼすべてのレジームを正しく抽出している．なお，DynaMMo はクラスタ発見の能力を持たない．

6.3　計算コスト

図 8 はシーケンスの長さ n を変化させた際の AutoPlait と比較手法における計算コストを示している．ここでは *MoCap* データを用いた．DynaMMo はパラメータとしてモデルの隠れ状態の個数を必要とするため，本実験では $k = 4$ とした．pHMM については $\epsilon_r = 0.1$, $\epsilon_c = 0.8$ とした．AutoPlait と DynaMMo はデータの長さに対し，線形 $O(n)$ である（対数スケールにおいて傾きは $slope = 1.0$ である）．一方，pHMM は $O(n^2)$ の計算量を要する（$slope \approx 2.0$）．AutoPlait は pHMM と比較し，$n = 100{,}000$ において 472 倍の性能向上を達成している．

7.　アプリケーション

本章では，AutoPlait の実用的なアプリケーション例を 2 つ紹介する．

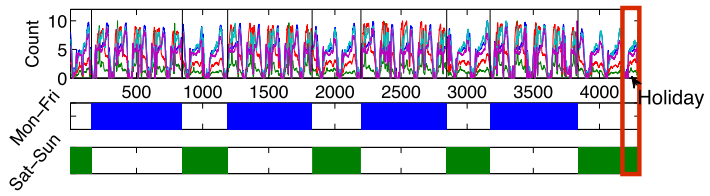
7.1　特徴抽出とモデル分析

AutoPlait は与えられた時系列データから特徴を抽出し，レジームの集合を確率モデルとして出力する．ここで得られる確率モデルは各レジームの時系列パターンを表現するため，データの統計的な分析を行うことができる．**図 9** は *WebClick* データにおける AutoPlait の出力結果を示している．具体的には，図 9 (a) は 1 カ月間における 5 つの主要な URL（blog や news 等）のアクセス数を 10 分ごとに記録したデータである．AutoPlait は，図のように，平日（青）と週末（緑）のレジームを発見している．ただし，月末の赤い枠で示した箇所は平日にもかかわらず例外的に週末のレジームの特徴を持つ．これは，この日が休日であることが原因である．図 9 (b)–(e) は，AutoPlait で推定された 2 つのレジーム（平日・週末）のモデルの詳細である．より具体的には，図 (b)，(c) はモデルのマルコフ連鎖（つまり隠れ状態の遷移）を表現し，図 (d)，(e) は各時間帯における 5 つの URL の出力確率を示している．両レジームともに，就寝時間帯にはほとんどアクセスがなく，次第にアクセスが増加し，午後 9 時にピークを迎えている．以下ではこれらのモデルから得られる知見をまとめる．

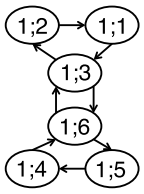
- 平日のレジームにおいて，状態 (1; 5)，(1; 6) は午前 10 時から午後 3 時に出現している．このことから，次の知見が得られる．(1) 多くのユーザが昼食休憩時にニュースサイトを閲覧している．(2) 大学生や社会人等が，日中オンライン辞書を頻繁に利用している．
- 週末の傾向は大幅に異なる．ブログ等のソーシャルメディアサイトは週末のアクセスがさかんである．一方で，ニュースや辞書等の仕事に関係するサイトは週末のアクセスが少ない．ただし例外として，平日においても，夕方から夜にかけてはブログやメールサイトへのアクセスがさかんである．

7.2　特徴的なイベントの自動検出

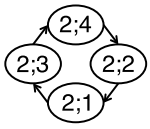
AutoPlait は時系列データの中から，未知のパターンと任意の数のレジームを自動的に発見することができる．ここでは，*GoogleTrend* データを使用して，Web 上のユーザ動向の情報を自動検出する例を考える．



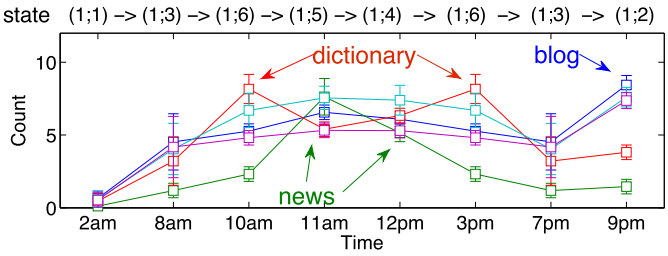
(a) 5 つの URL における 1 カ月間のパターン（blog，news，dictionary，Q&A，mail）



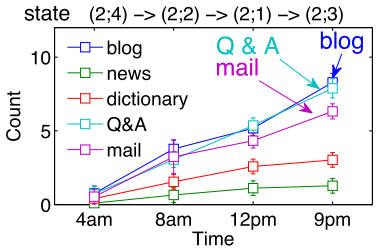
(b) 状態遷移確率 $\mathbf{A}_1$（平日，$k_1 = 6$）



(c) 状態遷移確率 $\mathbf{A}_2$（週末，$k_2 = 4$）



(d) 出力確率 $\mathbf{B}_1$（平日 レジーム）



(e) 出力確率 $\mathbf{B}_2$（週末 レジーム）

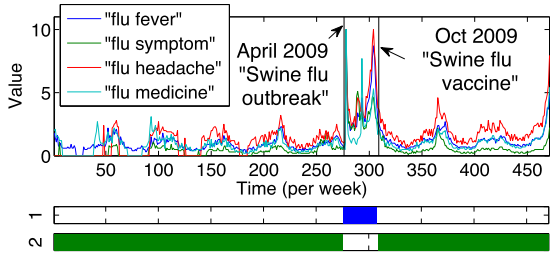
図 9　*WebClick* データにおける変化点抽出の結果 (a) と，得られたモデル（MLCM）の詳細 (b)–(e)

Fig. 9　Sense-making - AutoPlait results match intuition: (a) five time-sequences (blog, news, dictionary, etc.) on *WebClick* data (top), and the result of AutoPlait (bottom).

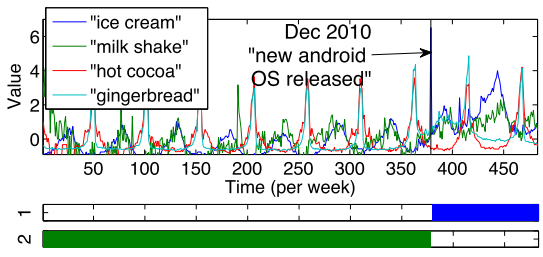
異常検出． 図 **10** (a) は，インフルエンザに関連するキーワード 4 つ（“flu fever”，“flu symptom” 等）の 9 年間の検索数を示している．このデータは年単位の周期性を持ち，毎年 10 月から 2 月にかけて検索数が増加し，次第に春，夏にかけて減少していく．この傾向は毎年同様であるが，2009 年は異なる傾向を持つ．これは，豚インフルエンザが世界的に大流行したことが原因である．AutoPlait はこの例外的なパターンをレジーム#1 として検出することに成功している．

変化点抽出． 図 10 (b) は，“ice cream”，“hot cocoa” のような季節のデザートに関連するクエリから生成したデータである．各キーワードは異なる位相を持ち，それぞれ年単位の周期を持つ．たとえば，“ice cream” と “milk shakes” は 7 月にピークを持ち，“hot cocoa” は “gingerbread” は 12 月に検索が集中する．しかし，このパターンは 2010 年の 12 月から変化していることが観察できる．これは，アンドロイドの OS である “Gingerbread” および “Ice Cream Sandwich” がリリースされた影響によるものである．

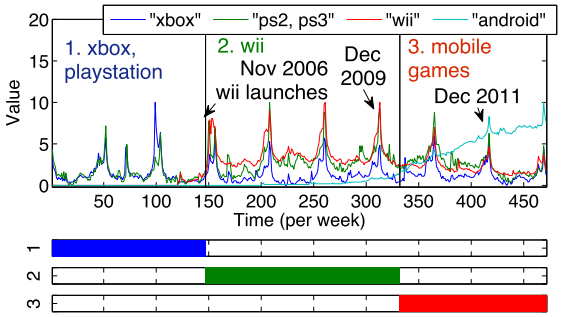
ドレンド発見． 図 10 (c) はゲーム産業に関連するキーワード（“xbox”，“wii” 等）の時系列データである．毎年 12 月のクリスマスにかけてピークを持つ．近年，ゲーム産業界は様々な企業の参入により，競争的になりつつある．AutoPlait は過去 9 年間のゲーム機戦争における 3 つのレジームを発見している．具体的には，(1) Xbox と Playstation が主流である時代から，(2) Wii が 2006 年に販売開始して以来の大幅なシェアの獲得，そして (3) 高性能携帯端末の普及によるモバイルゲームやソーシャルゲームの出現への流れをとらえている．



(a) インフルエンザ関連ワード（レジーム $r = 2$）



(b) 季節のデザート関連ワード（レジーム $r = 2$）



(c) ゲーム関連ワード（レジーム $r = 3$）

図 10　*GoogleTrend* データにおけるトレンドの変化点抽出例

Fig. 10　Sense-making: AutoPlait spots meaningful discontinuities.

8. むすび

本論文では大規模時系列データのための特徴自動抽出手法としてAutoPlaitを提案した．AutoPlaitは，ユーザによるパラメータ設定や事前知識を要することなく，与えられた大規模シーケンスに対し，ユーザの直感に合う複雑な時系列パターン（レジーム）とその変化点を発見することができる．また，セグメント分割位置については，最適解を出力することを保証している．様々な種類の実データを用いて実験を行い，AutoPlaitは最新の時系列解析手法と比べてより高い精度と性能を持つことを示した．

参考文献

[1] Böhm, C., Faloutsos, C. and Plant, C.: Outlier-robust clustering using independent components, *SIGMOD*, pp.185–198 (2008).

[2] Box, G.E., Jenkins, G.M. and Reinsel, G.C.: *Time Series Analysis: Forecasting and Control*, 3rd edition, Prentice Hall, Englewood Cliffs, NJ (1994).

[3] Chakrabarti, D., Papadimitriou, S., Modha, D.S. and Faloutsos, C.: Fully automatic cross-associations, *KDD*, pp.79–88 (2004).

[4] Chen, L. and Ng, R.T.: On the marriage of lp-norms and edit distance, *VLDB*, pp.792–803 (2004).

[5] Fine, S., Singer, Y. and Tishby, N.: The hierarchical hidden markov model: Analysis and applications, *Machine Learning*, Vol.32, No.1, pp.41–62 (1998).

[6] Fox, E.B., Sudderth, E.B., Jordan, M.I. and Willsky, A.S.: Sharing features among dynamical systems with beta processes, *NIPS*, pp.549–557 (2009).

[7] Fujiwara, Y., Sakurai, Y. and Yamamuro, M.: Spiral: Efficient and exact model identification for hidden markov models, *KDD*, pp.247–255 (2008).

[8] Jain, A., Chang, E.Y. and Wang, Y.-F.: Adaptive stream resource management using kalman filters, *SIGMOD*, pp.11–22 (2004).

[9] Keogh, E.J., Chu, S., Hart, D. and Pazzani, M.J.: An online algorithm for segmenting time series, *ICDM*, pp.289–296 (2001).

[10] Lee, J.-G., Han, J. and Whang, K.-Y.: Trajectory clustering: A partition-and-group framework, *SIGMOD Conference*, pp.593–604 (2007).

[11] Letchner, J., Ré, C., Balazinska, M. and Philipose, M.: Access methods for markovian streams, *ICDE*, pp.246–257 (2009).

[12] Li, L., McCann, J., Pollard, N. and Faloutsos, C.: Dynammo: Mining and summarization of coevolving sequences with missing values, *KDD* (2009).

[13] Li, L., Prakash, B.A. and Faloutsos, C.: Parsimonious linear fingerprinting for time series, *PVLDB*, Vol.3, No.1, pp.385–396 (2010).

[14] Matsubara, Y., Sakurai, Y., Faloutsos, C., Iwata, T. and Yoshikawa, M.: Fast mining and forecasting of complex time-stamped events, *KDD*, pp.271–279 (2012).

[15] Matsubara, Y., Sakurai, Y., Prakash, B.A., Li, L. and Faloutsos, C.: Rise and fall patterns of information diffusion: Model and implications, *KDD*, pp.6–14 (2012).

[16] Mueen, A. and Keogh, E.J.: Online discovery and maintenance of time series motifs, *KDD*, pp.1089–1098 (2010).

[17] Ng, R.T. and Han, J.: Clarans: A method for clustering objects for spatial data mining, *IEEE Trans. Knowl. Data Eng.*, Vol.14, No.5, pp.1003–1016 (2002).

[18] Palpanas, T., Vlachos, M., Keogh, E. and Gunopulos, D.: Streaming time series summarization using user-defined amnesic functions, *IEEE Trans. Knowl. Data Eng.*, Vol.20, No.7, pp.992–1006 (2008).

[19] Papadimitriou, S., Sun, J. and Faloutsos, C.: Streaming pattern discovery in multiple time-series, *Proc. VLDB*, Trondheim, Norway, August-September, pp.697–708 (2005).

[20] Rakthanmanon, T., Campana, B.J.L., Mueen, A., Batista, G.E.A.P.A., Westover, M.B., Zhu, Q., Zakaria, J. and Keogh, E.J.: Searching and mining trillions of time series subsequences under dynamic time warping, *KDD*, pp.262–270 (2012).

[21] Rissanen, J.: A Universal Prior for Integers and Estimation by Minimum Description Length, *Ann. of Statist.*, Vol.11, No.2, pp.416–431 (1983).

[22] Sakurai, Y., Faloutsos, C. and Yamamuro, M.: Stream monitoring under the time warping distance, *Proc. ICDE*, Istanbul, Turkey, April, pp.1046–1055 (2007).

[23] Sakurai, Y., Papadimitriou, S. and Faloutsos, C.: Braid: Stream mining through group lag correlations, *Proc. ACM SIGMOD*, Baltimore, Maryland, June, pp.599–610 (2005).

[24] Tao, Y., Faloutsos, C., Papadias, D. and Liu, B.: Prediction and indexing of moving objects with unknown motion patterns, *Proc. ACM SIGMOD*, pp.611–622 (2004).

[25] Tatti, N. and Vreeken, J.: The long and the short of it: Summarising event sequences with serial episodes, *KDD*, pp.462–470 (2012).

[26] Toyoda, M., Sakurai, Y. and Ishikawa, Y.: Pattern discovery in data streams under the time warping distance, *VLDB J.*, Vol.22, No.3, pp.295–318 (2013).

[27] Wang, P., Wang, H. and Wang, W.: Finding semantics in time series, *SIGMOD Conference*, pp.385–396 (2011).

[28] Wilpon, J.G., Rabiner, L.R., Lee, C.H. and Goldman, E.R.: Automatic recognition of keywords in unconstrained speech using hidden Markov models, *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol.38, No.11, pp.1870–1878 (1990).

[29] Zhang, T., Ramakrishnan, R. and Livny, M.: Birch: An efficient data clustering method for very large databases, *SIGMOD*, pp.103–114, ACM (1996).

松原 靖子

2006年お茶の水女子大学理学部情報科学科卒業．2009年同大学院人間文化創成科学研究科理学専攻博士前期課程修了．2012年京都大学大学院情報学研究科社会情報学専攻博士後期課程修了．博士（工学）．2012年NTTコミュニケーション科学基礎研究所RA．2013年より熊本大学大学院自然科学研究科，日本学術振興会特別研究員（PD）．この間，カーネギーメロン大学客員研究員．大規模時系列データマイニングに関する研究に従事．日本データベース学会会員．

櫻井 保志　（正会員）

1991 年同志社大学工学部電気工学科卒業．1991 年日本電信電話（株）入社．1999 年奈良先端科学技術大学院大学情報科学研究科博士後期課程修了．博士（工学）．2004～2005 年カーネギーメロン大学客員研究員．2013 年熊本大学大学院自然科学研究科教授．本会平成 18 年度長尾真記念特別賞，本会平成 16 年度および平成 19 年度論文賞，電子情報通信学会平成 19 年度論文賞，日本データベース学会上林奨励賞，ACM KDD best paper awards（2008，2010）等受賞．データマイニング，データストリーム処理，センサデータ処理，Web 情報解析技術の研究に従事．ACM，電子情報通信学会，日本データベース学会各会員．

Christos Faloutsos

Christos Faloutsos is a Professor at Carnegie Mellon University. He has received the Presidential Young Investigator Award by the National Science Foundation (1989), the Research Contributions Award in ICDM 2006, the SIGKDD Innovations Award (2010), seventeen best paper awards (including two ‘test of time’), and four teaching awards. He has served as a member of the executive committee of SIGKDD; he is an ACM Fellow; he has published over 200 refereed articles, 11 book chapters, and one monograph. He holds five patents and he has given over 30 tutorials and over 10 invited distinguished lectures. His research interests include data mining for graphs and streams, fractals, database performance, and indexing for multimedia and bioinformatics data.

（担当編集委員　平手 勇宇）