社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2009－NL－190 (16)
2009／3／26

系列ラベリングのための前向き後ろ向きアルゴリズムの一般化

東 藍[†1] 松本 裕治[†1]

本稿では，前向き後ろ向きアルゴリズムの一般化について述べる．既存の系列ラベリングに関する研究のいくつかでは，従来の前向き後ろ向きアルゴリズムでは取り扱えない種類の計算を必要としていたが，本稿で述べる一般化により，そのような計算の多くも取り扱えるようになることを示す．

A Generalization of Forward-backward Algorithm for Sequential Labeling

Ai Azuma[†1] and Yuji Matsumoto[†1]

In this paper, we propose a generalization of Forward-backward algorithm. In existing research, it is required to calculate values with slightly complex forms. Ordinary Forward-backward algorithm cannot treat them. The generalizing method we propose can treat such calculations.

1. はじめに

系列ラベリングとして帰着できる自然言語処理の問題は多い．本稿では，系列ラベリングに関する既存の研究のいくつかにおいて従来の前向き後ろ向きアルゴリズムでは取り扱えない種類の計算が必要とされていることを，実例を挙げて指摘する．そして，それらの計算において必要とされているものを汎用な形で定式化し，その形式に対応できるような前向き後ろ向きアルゴリズムの一般化を提示する．実験では，従来の前向き後ろ向きアルゴリズムでは取り扱えない計算に関して，実際の計算精度と計算量の挙動について実証した．

2. 研究の動機

自然言語処理のタスクの多くは，構造学習，すなわちある構造を入力としその入力構造に対してもっともらしい出力構造を推定する問題としてモデルできる．入力を文字列とし出力を形態素列 (単語列) とする形態素解析，入力を文節列とし出力を係り受け木とする係り受け解析など枚挙に暇が無い．

構造学習が対象とする構造のうち，最も基本的なものは系列である．最も基本的な構造ではあるが，自然言語処理におけるその応用は広く，形態素解析，固有表現抽出，文節同定などに利用される．これらのタスクは入力列に対してもっともらしい正解ラベル列を付与する問題，いわゆる系列ラベリングとして定式化される．

†1 奈良先端科学技術大学院大学
Nara Institute of Science and Technology

系列ラベリングにおいては，特に確率モデルとして隠れマルコフモデル (Hidden Markov Models; HMM)，あるいは条件付確率場 (Conditional Random Fields; CRF)[5] などを仮定し，大規模な正解タグ付きデータに対する最尤推定などの枠組みでモデルパラメタを推定する統計学習的手法が広く定着している．

さて，HMM に対する EM アルゴリズム[2] においては，現在のモデルにおける隠れ変数のモデル期待値を計算する手順が必要となる．また，CRF に対する条件付対数尤度最大化，あるいは事後確率最大化に基づくモデルパラメタ推定において，特に勾配法に基づく非線型最適化として解く際には，分配関数の計算，およびパラメタ偏微分の算出に必要となる素性関数のモデル期待値の計算が必要である．ここで挙げた計算は，与えられたある入力系列に対して可能な全ての出力候補系列に対するある種の総和として定義される．一般に出力候補系列の数は膨大――典型的には入力系列の長さに対して指数関数オーダー――である．したがって，この計算を定義どおりに実行することは実質的に不可能である．

この計算を実行するにあたり，与えられた入力系列に対して可能な全ての出力候補系列を陽に列挙することなく効率的に行う手法が広く知られている．前向き後ろ向きアルゴリズムと呼ばれるアルゴリズムがそれである．このアルゴリズムの存在は上記の手法を実現するための 1 つの重要な要素である．

ここで，可能な全ての出力候補系列に対する総和を計算するアルゴリズムが「効率的である」とは，可能な全ての出力候補系列の数よりも十分小さい時間計算量・空間計算量でアルゴリズムが実行可能であること

を指すものとする．以下，本稿で「効率的である」という表現を用いる際にはこの意味で用いるものとする．

前向き後ろ向きアルゴリズムを，可能な全ての出力候補系列に関してある種の総和を計算するアルゴリズムとして認識するならば，その「ある種」とは極めて限られた種類の形式であるといわざるを得ない．すなわち，前向き後ろ向きアルゴリズムがその対象とするのは

$$\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \Phi(\mathbf{x},\mathbf{y}) \tag{1}$$

あるいは

$$\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \Phi(\mathbf{x},\mathbf{y})F(\mathbf{x},\mathbf{y}) \tag{2}$$

という形式で定義される値の計算に限られる．ここで $\mathcal{Y}(\mathbf{x})$ は入力系列 $\mathbf{x}$ に対して可能なすべての出力候補系列の集合を表すものとする．また，$\Phi(\mathbf{x},\mathbf{y})$ は系列中のラベルまたは隣接するラベルペア上に定義された関数の積として因数化されている必要がある．すなわち，

$$\Phi(\mathbf{x},\mathbf{y}) := \prod_{c\in\mathcal{C}(\mathbf{y})} \phi(\mathbf{x},c) \tag{3}$$

なる形式である必要がある．$\mathcal{C}(\mathbf{y})$ は，系列中のラベルの集合と系列中の隣接ラベルペアの集合との和集合を表す．さらに，$F(\mathbf{x},\mathbf{y})$ に関しても $\mathcal{C}(\mathbf{y})$ 上に定義された関数の和として定義されている必要がある．つまり，

$$F(\mathbf{x},\mathbf{y}) := \sum_{c\in\mathcal{C}(\mathbf{y})} f(\mathbf{x},c) \tag{4}$$

なる形式である必要がある．以下では，表記を簡潔にするため式中で明らかな $\mathbf{x}$ は省略する．すなわち，$\Phi(\mathbf{x},\mathbf{y})$ や $F(\mathbf{x},\mathbf{y})$ などを，各々$\Phi(\mathbf{y})$ や $F(\mathbf{y})$ などと表記することにする．

近年においては，系列ラベリングを適用するタスクの状況に応じた様々な確率モデル，パラメタ推定法，最適化手法等々が提案されている．これらの手法においてはやはり，可能な全ての出力候補系列に対するある種の総和を計算する必要がしばしば生じる．これらの計算の多くは前向き後ろ向きアルゴリズムで計算できる形式 (1)(2) を逸脱している．したがって，効率的に値を計算するための様々な算法や近似が個々の問題の状況に応じて考案されてきた．

たとえば [Jiao et al., 2006][4] では，CRF に対する半教師付学習の枠組みにおいてエントロピに対するパラメタ偏微分を必要とする．エントロピ

$$\mathcal{H}(\mathbf{y}|\mathbf{x};\lambda) := E_{P(\mathbf{y}|\mathbf{x};\lambda)}[\log P(\mathbf{y}|\mathbf{x};\lambda)] \tag{5}$$

の定義から，CRF におけるエントロピのパラメタ偏微分は

$$\begin{aligned}\frac{\partial}{\partial\lambda}\mathcal{H}(\mathbf{y}|\mathbf{x};\lambda) &= -\operatorname{Cov}_{P(\mathbf{y}|\mathbf{x};\lambda)}[\mathbf{F}(\mathbf{y}),\mathbf{F}(\mathbf{y})]\lambda \\ &= E_P[\mathbf{F}(\mathbf{y})]E_P[\mathbf{F}(\mathbf{y})\cdot\lambda] \\ &\quad - E_P[\mathbf{F}(\mathbf{y})(\mathbf{F}(\mathbf{y})\cdot\lambda)]\end{aligned} \tag{6}$$

($\mathbf{F}(\mathbf{y})$ は大域素性ベクトル) となる．(6) の最終辺第 2 項は前向き後ろ向きアルゴリズムで計算できない形式の総和を要する．なお，[Mann et al., 2007][7] では，前向き後ろ向きアルゴリズムを実行するために必要となる計算量の高々定数倍で (6) の値を計算できるアルゴリズムをエントロピの性質に依存した形で導出している．

[Vishwanathan et al., 2006][9] では確率的降下法を CRF におけるパラメタ推定に適用している．この際に対数尤度関数のヘッセ行列と任意のベクトルとの積を計算する必要が生じる．CRF の対数尤度関数に対するヘッセ行列 $\mathbf{H}(\mathcal{L})$ が

$$\mathbf{H}(\mathcal{L}) = -\operatorname{Cov}_{P(\mathbf{y}|\mathbf{x};\lambda)}[\mathbf{F}(\mathbf{y}),\mathbf{F}(\mathbf{y})] \tag{7}$$

($\mathbf{F}(\mathbf{y})$ は大域素性ベクトル) となることから，

$$\begin{aligned}\mathbf{H}(\mathcal{L})\cdot\mathbf{v} &= E_P[\mathbf{F}(\mathbf{y})]E_P[\mathbf{F}(\mathbf{y})\cdot\mathbf{v}] \\ &\quad - E_P[\mathbf{F}(\mathbf{y})(\mathbf{F}(\mathbf{y})\cdot\mathbf{v})]\end{aligned} \tag{8}$$

である．(8) の右辺第 2 項は前向き後ろ向きアルゴリズムでは計算できない種類の総和を必要とする．彼らは，(8) の計算を，対数尤度関数に対するパラメタ偏微分を算出するアルゴリズムに対して自動微分法を適用することによって実行している．これはヘッセ行列とベクトルとの積を計算する目的に特化した手法である．

上に述べたように，CRF に対するエントロピのパラメタ偏微分，および CRF の対数尤度関数に対するヘッセ行列-ベクトル積の各計算は，前向き後ろ向きアルゴリズムで計算できない種類の総和を含む．これらの総和計算は以下の形式に一般化して書き下すことができよう．

$$\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \Phi(\mathbf{y})F_1(\mathbf{y})F_2(\mathbf{y}) \tag{9}$$

ただし，ここで $\Phi(\mathbf{y})$ は (3) の条件を満たすものとし，F_1, F_2 は (4) の条件同様

$$F_k(\mathbf{y}) := \sum_{c\in\mathcal{C}(\mathbf{y})} f_k(c) \quad (k=1,2) \tag{10}$$

なる形式に限るものとする．さて，ここで——やや数学的な興味ながら——次のような疑問が生じる．すなわち，$\Phi(\mathbf{y}), F_1(\mathbf{y}), F_2(\mathbf{y})$ に対して (3) (10) 以外の条件に一切依存することなく，(9) を効率的に計算するようなアルゴリズムは存在するであろうか? 上で引用した文献では，各々の計算対象において特異に付随する性質を用いて効率的なアルゴリズムを導出していることに留意されたい．また，(1), (2), (9) と，計算対象に含まれる関数の数が増えていくところから，より一般的に，K 個 ($K\in\mathbb{N}$) の関数 $F_k(\mathbf{y})$ $(k=1,\ldots,K)$ が

$$F_k(\mathbf{y}) := \sum_{c\in\mathcal{C}(\mathbf{y})} f_k(c) \quad (k=1,\ldots,K) \tag{11}$$

なる条件を満たすとき，

$$\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \Phi(\mathbf{y})F_1(\mathbf{y})\cdots F_K(\mathbf{y}) \tag{12}$$

なる和が効率的に計算可能であろうか，という疑問に思い至るのは極めて自然であろう．

この疑問に対する本稿における回答は YES であり，(12) を効率的に計算するアルゴリズムを実際に導出

する．

さて，本稿で提案するアルゴリズムは (12) よりもさらに一般化した形式で与える．このさらなる一般化の動機を説明するために，可能な全ての出力候補系列に対してさらに複雑な形式の総和計算を実行する要求が生じる例を以下に加えよう．

[Altun et al., 2004][1], [Lafferty et al., 2004][6] では指数分布族に対してカーネル法を適用する提案を行っている．モデルの妥当性，再生核の理論などの詳細は引用先の記述に譲るとして，ここではその確率密度関数の形式のみに焦点を当てる．カーネル関数 $\mathcal{K}(\cdot,\cdot)$ を導入した指数分布族は次の形式を持つ．

$$P(\mathbf{y}|\mathbf{x};\lambda) = \exp(\mathcal{K}(\lambda, \mathbf{F}(\mathbf{y})) - A(\lambda;\mathbf{x})) \quad (13)$$

これはカーネル関数 $\mathcal{K}$ が定める再生核ヒルベルト空間へ $\mathbf{F}(\mathbf{y})$ を写像した上で指数分布族を仮定していることに対応する．ここで

$$\begin{aligned} Z(\lambda;\mathbf{x}) &:= \exp(A(\lambda;\mathbf{x})) \\ &= \sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \exp(\mathcal{K}(\lambda, \mathbf{F}(\mathbf{y})) \end{aligned} \quad (14)$$

は分配関数である．今，たとえば $\mathcal{K}$ が 2 次以上の多項式カーネルである場合を想定しよう．この場合 $\exp(\mathcal{K}(\lambda, \mathbf{F}(\mathbf{y})))$ はもはや $\mathbf{y}$ 中のラベル・隣接ラベルペアに関する関数の積として因数化できない．この事実は，(14) の計算が前向き後ろ向きアルゴリズムの取り扱える範疇にないことを意味する．

[Jansche, 2005][3] では，ロジスティック回帰モデルに対してラベル毎の F-値の期待値を最大化するようなパラメタを学習の目標とすることを提案している．今，これを愚直に系列ラベリングへ拡張することを考えよう．系列ラベリングに対するラベル毎の F 値は以下のように定義される．

$$F_\gamma(\acute{\mathbf{y}}) := \frac{(1+\gamma^2)l(\acute{\mathbf{y}})}{|\acute{\mathbf{y}}| + \gamma^2|\hat{\acute{\mathbf{y}}}|} \quad (15)$$

ここで $\acute{\mathbf{y}} \in \bigotimes_i \mathcal{Y}(\mathbf{x}_i)$ は，入力系列 $\mathbf{x}_1$ に対して可能なある出力候補系列 $\mathbf{y}_1$ の後ろに，入力系列 $\mathbf{x}_2$ に対して可能なある出力候補系列 $\mathbf{y}_2$ をつなぎ，以下同様にある出力候補系列をすべてつなぎ合わせて作られるある系列である．$|\hat{\acute{\mathbf{y}}}| := \sum_i |\hat{\mathbf{y}}_i|$ は正解タグ付き学習データ全てで正解出力ラベル数を総和したもの，$l(\acute{\mathbf{y}})$ は $\acute{\mathbf{y}}$ 中の正解ラベルの数である．今，確率モデル $P(\mathbf{y}|\mathbf{x};\lambda)$ として CRF を仮定すると，この関数に対するモデル期待値は

$$E_{P(\acute{\mathbf{y}}|\acute{\mathbf{x}};\lambda)}[F_\gamma(\acute{\mathbf{y}})] = \frac{1}{Z(\acute{\mathbf{x}};\lambda)} \sum_{\acute{\mathbf{y}}} \Phi(\acute{\mathbf{y}}) F_\gamma(\acute{\mathbf{y}}) \quad (16)$$

となる．ここで $Z(\acute{\mathbf{x}};\lambda)$ は分配関数であり，$\Phi(\acute{\mathbf{y}})$ は (3) の形式を取る関数である．また，これに対するパラメタ偏微分

$$\begin{aligned} &\frac{\partial}{\partial\lambda} E_{P(\acute{\mathbf{y}}|\acute{\mathbf{x}};\lambda)}[F_\gamma(\mathbf{y})] \\ &\quad = -\operatorname{Cov}_{P(\acute{\mathbf{y}}|\acute{\mathbf{x}};\lambda)}[F_\gamma(\acute{\mathbf{y}}), \mathbf{F}(\acute{\mathbf{y}})] \\ &\quad = E_P[F_\gamma(\acute{\mathbf{y}})]E_P[\mathbf{F}(\acute{\mathbf{y}})] - E_P[F_\gamma(\acute{\mathbf{y}})\mathbf{F}(\acute{\mathbf{y}})] \end{aligned} \quad (17)$$

を計算すれば，パラメタの推定は勾配法に基づく非線型最適化に帰着できる．さて，$^\forall\acute{\mathbf{y}}$ に対してその長さ $|\acute{\mathbf{y}}|$ が同一である場合は，$|\acute{\mathbf{y}}|$ は $\sum_{\acute{\mathbf{y}}}$ の和において定数であり，また $l(\acute{\mathbf{y}})$ は (4) の形式を取る．この場合は，(16) の値と (17) の最終辺第 1 項の値は従来の前向き後ろ向きアルゴリズムで計算でき，また，(17) の最終辺第 2 項は (9) の形式となる．一方で，Semi-markov モデル[8] のように出力候補の長さ $|\acute{\mathbf{y}}|$ が可変の場合，(16) および (17) を効率的に計算する手法はまったく自明ではない．

さて，上に挙げたカーネル付指数分布族の推定や期待 F 値を目標関数とするパラメタ推定で必要となる計算は (12) の形式の範疇ですらない．(4) を満たすような関数 $F(\mathbf{y})$ にさらに非線型関数を適用した形式に対して総和を取らねばならないのである．しかしながら，たとえば指数分布に 2 次多項式カーネルを導入した場合の分配関数，

$$\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \exp((\lambda^T\mathbf{F}(\mathbf{y}) + c)^2) \quad (c\in\mathbb{R}) \quad (18)$$

について考えてみよう．これはテイラー展開すると

$$\begin{aligned} &\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \exp((\lambda^T\mathbf{F}(\mathbf{y}) + c)^2) \\ &\quad = \sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \sum_{m=0}^{\infty} \frac{(\lambda^T\mathbf{F}(\mathbf{y}) + c)^{2m}}{m!} \\ &\quad = \sum_{m=0}^{\infty} \frac{1}{m!} \sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} (\lambda^T\mathbf{F}(\mathbf{y}) + c)^{2m} \end{aligned} \quad (19)$$

となる．このテイラー展開を適当な次数で打ち切るとすると，それはそのまま (18) の近似計算アルゴリズムとなる．ここで $\lambda^T\mathbf{F}(\mathbf{y}) + c$ は (4) を満たす形式に書き下せることに特に留意されたい．

ここに至ってさらなる一般化の動機を得る．すなわち，条件 (4) を満たす関数 $F(\mathbf{y})$ の自然数べきを可能なすべての出力候補系列に関して総和したい，というものである．形式的に書き下せば，[*1]

$$\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \Phi(\mathbf{y})F^m(\mathbf{y}) \quad (m = 0,1,2,\dots) \quad (20)$$

を効率的に計算するアルゴリズムは存在するであろうか，という疑問になる．無論，(12) に対して $F_k(\mathbf{y}) := F(\mathbf{y}) \quad (k = 1,\dots,m)$ とおけば (20) の計算は (12) の形式に帰着される．しかしながら，後に明らかになるが，(12) の形式に帰着するよりも効率的にべき乗の総和を計算できる形式でアルゴリズムを与える．

以上の総括として，本稿の中心となる目標を最も一般的な形式で述べよう．$n_1,\dots,n_K \in \mathbb{N}_0$ ($\mathbb{N}_0$ は非負整数の集合) に対して，

$$\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})} \Phi(\mathbf{y}){F_1}^{n_1}(\mathbf{y})\cdots{F_K}^{n_K}(\mathbf{y}) \quad (21)$$

が効率的に計算できるアルゴリズムが導出できれば，個々の計算の状況に特化しない，きわめて汎用なアル

[*1] 本稿では表記上 0 の 0 乗が現れうる．この際は便宜上 $0^0 := 1$ と定義する．

ゴリズムを提供できることとなる．そして，本稿ではそのようなアルゴリズムを実際に導出する．

3. 前向き後ろ向きアルゴリズムの一般化

本節では一般化した前向き後ろ向きアルゴリズムを定式化する．定義・証明を行う際の表記の都合上，$\mathcal{Y}(\mathbf{x})$ が形成する構造 (ラティス) を非循環有向グラフ (Directed Acyclic Graph; DAG) に置き換える．つまり，ある出力候補系列 $\mathbf{y} \in \mathcal{Y}(\mathbf{x})$ 中のラベルを DAG G のある有向経路中の頂点に，$\mathbf{y}$ 中の隣接ラベルペアをその有向経路中の辺に対応させることで，結果としてある出力候補系列 $\mathbf{y}$ を G 中のある有向経路として表現することにする．以下，G は入次数 0 の頂点をただ 1 つだけ持つ (これを $\mathrm{src}(G)$ とおく．混乱の恐れが無い文脈では src と略す) ものとし，かつ出次数 0 の頂点もただ 1 つだけ持つ (これを $\mathrm{snk}(G)$ とおく．混乱の恐れが無い文脈では snk と略す) ものとする．入次数 0 の頂点を複数持つ場合，または出次数 0 の頂点を複数持つ場合への拡張は極めて平易であるため以下では省略する．$\mathrm{src}(G)$ を始点とし $\mathrm{snk}(G)$ を終点とする有向経路の集合を $\Psi(G)$ と表記する．以降，$\mathcal{Y}(\mathbf{x})$ の各要素は $\Psi(G)$ の各要素に 1 対 1 対応するものとして議論を進める．以上の置き換えにより，出力候補系列に関する和 $\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})}(\cdot)$ は，$\Psi(G)$ に対する和 $\sum_{\pi\in\Psi(G)}(\cdot)$ として定式化できる．

3.1 標準的な形式

定理．(一般化前向きアルゴリズム)

DAG $G=(V,E)$ を考える．表記の簡略化のために，$\Psi(G)$ の要素 π をそこに含まれる頂点の集合を表記するものとしても扱うことにする．V 上に定義された K 個の複素数値関数 $f_k(v)\quad(k=1,\ldots,K)$ があり，さらに $\Psi(G)$ 上の関数 $F_k(\pi):=\sum_{v\in\pi}f_k(v)\quad(k=1,\ldots,K)$ を定義する．同じく，V 上に定義された複素数値関数 $\phi(v)$ があり，さらに $\Psi(G)$ 上の関数 $\Phi(\pi):=\prod_{v\in\pi}\phi(v)$ を定義する．[⋆1] このとき $n_1,\ldots,n_K\in\mathbb{N}_0$ に対して以下が成り立つ．

$$\sum_{\pi\in\Psi(G)}\Phi(\pi){F_1}^{n_1}(\pi)\cdots{F_K}^{n_K}(\pi)=\alpha_{n_1,\ldots,n_K}(\mathrm{snk}) \tag{22}$$

ただし $\alpha_{n_1,\ldots,n_K}(v)$ は，G 上のある topological order $(V,\prec)$ に沿って以下のように再帰的に定義される．

$$\alpha_{n_1,\ldots,n_K}(\mathrm{src}):=\phi(\mathrm{src}){f_1}^{n_1}(\mathrm{src})\cdots{f_K}^{n_K}(\mathrm{src}),$$

$$\begin{aligned}&\alpha_{n_1,\ldots,n_K}(v)\\&:=\phi(v)\sum_{m_1=0}^{n_1}\left[\binom{n_1}{m_1}{f_1}^{m_1}(v)\cdots\right.\\&\quad\left.\times\sum_{m_K=0}^{n_K}\left[\binom{n_K}{m_K}{f_K}^{m_K}(v)\right.\right.\\&\quad\left.\left.\times\sum_{v'\in\mathrm{prev}(v)}\alpha_{n_1-m_1,\ldots,n_K-m_K}(v')\right]\cdots\right]\\&\qquad(v\neq\mathrm{src})\end{aligned} \tag{23}$$

ここで $\binom{n}{m}$ は 2 項係数，$\mathrm{prev}(v)$ は v の直前の頂点の集合 $\mathrm{prev}(v):=\{v'|(v',v)\in E\}$ である．

証 明 ．

以下では，α の再帰的定義 (23) に基づいて (24) を証明する．

$$\alpha_{n_1,\ldots,n_K}(v)=\sum_{\pi\in\Psi_{\mathrm{src}\to v}(G)}\Phi(\pi){F_1}^{n_1}(\pi)\cdots{F_K}^{n_K}(\pi) \tag{24}$$

ここで $\Psi_{\mathrm{src}\to v}(G)$ は，src を始点とし $v\in V$ を終点とする有向経路の集合である．証明は $(V,\prec)$ に沿って帰納的に行う．

$v=\mathrm{src}$ のとき，(24) は α の定義 (23) から自明．

以下，簡単のために α の添え字が 1 つ $n=n_1$ $(f=f_1,F=F_1)$ だけの場合について証明を進める．

ある $v\in V$ のすべての直前の頂点 $^\forall v'\in\mathrm{prev}(v)$ に対して (24) の成立を仮定すると，

$$\begin{aligned}&\alpha_n(v)\\&=\phi(v)\sum_{m=0}^{n}\left[\binom{n}{m}f^m(v)\sum_{v'\in\mathrm{prev}(v)}\alpha_{n-m}(v')\right]\\&\qquad(\because\alpha\text{ の定義 (23) から})\\&=\phi(v)\sum_{m=0}^{n}\left[\binom{n}{m}f^m(v)\right.\\&\quad\left.\times\sum_{v'\in\mathrm{prev}(v)}\sum_{\pi\in\Psi_{\mathrm{src}\to v'}(G)}\Phi(\pi)F^{n-m}(\pi)\right]\\&(\because{}^\forall v'\in\mathrm{prev}(v)\text{ に対する帰納法の仮定 (24) から})\\&=\sum_{v'\in\mathrm{prev}(v)}\sum_{\pi\in\Psi_{\mathrm{src}\to v'}(G)}\left[\phi(v)\Phi(\pi)\right.\\&\quad\left.\times\sum_{m=0}^{n}\binom{n}{m}f^m(v)F^{n-m}(\pi)\right]\\&=\sum_{v'\in\mathrm{prev}(v)}\sum_{\pi\in\Psi_{\mathrm{src}\to v'}(G)}\phi(v)\Phi(\pi)(f(v)+F(\pi))^n\\&\qquad(\because\text{2 項定理})\\&=\sum_{v'\in\mathrm{prev}(v)}\sum_{\pi\in\Psi_{\mathrm{src}\to v'}(G)}\Phi(\pi\cup\{v\})(F(\pi\cup\{v\}))^n\\&=\sum_{\pi\in\Psi_{\mathrm{src}\to v}(G)}\Phi(\pi)F^n(\pi)\end{aligned} \tag{25}$$

となり，v に対して (24) が成立することが分かる．したがって $^\forall v\in V$ に対して (24) が成立することが帰納的に示される．α の添え字が複数の場合，すなわち $\alpha_{n_1,\ldots,n_K}$ に対してもまったく同様の証明で $^\forall v\in V$ に対して (24) が示され，これから直ちに (22) が示される．□

⋆1 記述を簡潔にするため f_k や ϕ が頂点に対してだけ定義されている場合のみ述べているが，辺に対してこれらの定義を拡張しても以降の記述はまったく同様に成り立つ．

(23) の再帰計算を用いて (22) を求めるアルゴリズムを $(n_1,\ldots,n_K)$-次の前向きアルゴリズムと呼ぶことにする．特に 0-次の前向きアルゴリズムは従来の前向き後ろ向きアルゴリズムにおける前向きの再帰計算と一致する．

ここで，ベクトル $\hat{\mathbf{F}}(\pi) := \sum_{v\in\pi} \hat{\mathbf{f}}(v)$ に関して，

$$\sum_{\pi\in\Psi(G)} \Phi(\pi){F_1}^{n_1}(\pi)\cdots{F_K}^{n_K}(\pi)\hat{\mathbf{F}}(\pi) \qquad (26)$$

なる和を計算する場合を考える．このような計算もまた様々な文脈で必要となり，2 節で挙げた各々の例でも必要となっていることが分かる．(26) で定義されるベクトル値は，$(n_1,\ldots,n_K,1)$-次の前向きアルゴリズムを高々 $\hat{\mathbf{F}}(\pi)$ の次元数だけ繰り返せば計算できることは明らかである．しかしながら，$\hat{\mathbf{F}}(\pi)$ が $|V|+|E|$ よりも有意に大きい値の次元数を持つベクトルであり，かつ，$\hat{\mathbf{f}}(v)$ が疎なベクトルであるならば，$(n_1,\ldots,n_K,1)$-次の前向きアルゴリズムを繰り返して (26) の値を計算するよりも，次に述べる形式のアルゴリズムのほうが計算の効率上有利になる可能性がある．

定理．(一般化前向き後ろ向きアルゴリズム)

一般化前向きアルゴリズムの定理を言明した際に用いた表記を流用する．加えて，V 上に複素数値関数 $\acute{f}(v)$ が定義されているものとし，さらに $\acute{F}(\pi) := \sum_{v\in\pi}\acute{f}(v)$ とする．このとき，

$$\sum_{\pi\in\Psi(G)} \Phi(\pi){F_1}^{n_1}(\pi)\cdots{F_K}^{n_K}(\pi)\acute{F}(\pi)$$
$$= \sum_{v\in V}\left[\acute{f}(v)\sum_{m_1=0}^{n_1}\left[\binom{n_1}{m_1}\cdots\sum_{m_K=0}^{n_K}\left[\binom{n_K}{m_K}\right.\right.\right.$$
$$\left.\left.\left.\times\,\alpha_{m_1,\ldots,m_K}(v)\beta_{n_1-m_1,\ldots,n_K-m_K}(v)\right]\cdots\right]\right] \qquad (27)$$

が成り立つ．ここで $\beta_{n_1,\ldots,n_K}(v)$ は，以下のように $(V,\prec)$ の逆順で再帰的に定義される．

$$\beta_{n_1,\ldots,n_K}(\mathrm{snk}) := \begin{cases} 1 & (n_1,\ldots,n_K=0),\\ 0 & (\text{otherwise}) \end{cases}$$

$$\beta_{n_1,\ldots,n_K}(v)$$
$$:= \sum_{v'\in\mathrm{next}(v)}\left[\phi(v')\sum_{m_1=0}^{n_1}\left[\binom{n_1}{m_1}{f_1}^{m_1}(v')\cdots\right.\right.$$
$$\left.\left.\times\sum_{m_K=0}^{n_K}\binom{n_K}{m_K}{f_K}^{m_K}(v')\beta_{n_1-m_1,\ldots,n_K-m_K}(v')\right]\right] \qquad (28)$$

また，$\mathrm{next}(v)$ は v の直後の頂点の集合 $\mathrm{next}(v) := \{v'|(v,v')\in E\}$ を表すものとする．

証 明 .

$\Psi_{v\leftarrow\mathrm{snk}}(G)$ を，v から snk に至る各有向経路から各々 v を除いたようなものの集合とする．すなわち $\Psi_{v\leftarrow\mathrm{snk}}(G) := \{\pi\setminus\{v\}|\pi\in\Psi_{v\to\mathrm{snk}}(G)\}$ である．α と同様，β に対して

$$\beta_{n_1,\ldots,n_K}(v) = \sum_{\pi\in\Psi_{v\leftarrow\mathrm{snk}}(G)}\Phi(\pi){F_1}^{n_1}(\pi)\cdots{F_K}^{n_K}(\pi) \qquad (29)$$

が $^\forall v\in V$(ただし snk を除く) で成立することを示すことが出来る．ここで，$\Psi(G)$ 上に定義された任意の関数 $\Gamma(\pi)$ に対して，和の交換

$$\sum_{\pi\in\Psi(G)}\Gamma(\pi)\acute{F}(\pi) = \sum_{v\in V}\left[\acute{f}(v)\sum_{\pi\in\Psi_v(G)}\Gamma(\pi)\right] \qquad (30)$$

が成り立つことに留意する．ただし，$\Psi_v(G)$ は $\Psi(G)$ の要素のうち v を通るようなものの集合である．明らかに $\Psi_v(G)\leftrightarrow\Psi_{\mathrm{src}\to v}(G)\otimes\Psi_{v\leftarrow\mathrm{snk}}(G)$ である．以下，簡単のために，α と β の添え字が 1 つ $n=n_1$ $(f=f_1, F=F_1)$ の場合だけに限って証明を進めると，

$$\sum_{\pi\in\Psi(G)}\Phi(\pi)F^n(\pi)\acute{F}(\pi)$$
$$= \sum_{v\in V}\left[\acute{f}(v)\sum_{\pi\in\Psi_v(G)}\Phi(\pi)F^n(\pi)\right]$$
$$(\because \Gamma(\pi) := \Phi(\pi)F^n(\pi)\text{ として (30) を適用})$$
$$= \sum_{v\in V}\left[\acute{f}(v)\sum_{\pi\in\Psi_{\mathrm{src}\to v}(G)}\left[\right.\right.$$
$$\left.\left.\times\sum_{\pi'\in\Psi_{v\leftarrow\mathrm{snk}}(G)}\Phi(\pi\cup\pi')F^n(\pi\cup\pi')\right]\right]$$
$$= \sum_{v\in V}\left[\acute{f}(v)\sum_{\pi\in\Psi_{\mathrm{src}\to v}(G)}\sum_{\pi'\in\Psi_{v\leftarrow\mathrm{snk}}(G)}\left[\Phi(\pi)\Phi(\pi')\right.\right.$$
$$\left.\left.\times\sum_{m=0}^{n}\binom{n}{m}F^m(\pi)F^{n-m}(\pi')\right]\right]$$
$$(\because 2\text{ 項定理})$$
$$= \sum_{v\in V}\left[\acute{f}(v)\sum_{m=0}^{n}\left[\binom{n}{m}\left(\sum_{\pi\in\Psi_{\mathrm{src}\to v}(G)}\Phi(\pi)F^m(\pi)\right)\right.\right.$$
$$\left.\left.\times\left(\sum_{\pi'\in\Psi_{v\leftarrow\mathrm{snk}}(G)}\Phi(\pi')F^{n-m}(\pi')\right)\right]\right]$$
$$= \sum_{v\in V}\left[\acute{f}(v)\sum_{m=0}^{n}\binom{n}{m}\alpha_m(v)\beta_{n-m}(v)\right]$$
$$(\because (24),(29)) \qquad (31)$$

となり，題意が示される．一般の場合である (27) もまったく同様に証明できる． □

前向きの再帰 (23) による α の計算，および後ろ向きの再帰 (28) による β の計算をもとに (27) を計算するアルゴリズムを $(n_1,\ldots,n_K)$-次の前向き後ろ向きアルゴリズムと呼ぶことにする．特に 0-次の前向き後ろ向きアルゴリズムは従来の前向き後ろ向きアルゴリズムと等しい．

3.2 フーリエ変換に基づく形式

以上に述べたアルゴリズムに加え，フーリエ変換と畳み込み定理を利用してより高速な形のアルゴリズム

が導出できることを示す．以下，簡単のため α の添え字が 1 つだけ $n = n_1$ の場合についてのみ論じる．

(23) における α の定義において，2 項係数の定義 $\binom{n}{m} := \frac{n!}{m!(n-m)!}$ を代入して変形すると以下を得る．

$$\frac{\alpha_n(\mathrm{src})}{n!} = \frac{f^n(\mathrm{src})}{n!},$$
$$\frac{\alpha_n(v)}{n!} = \phi(v) \sum_{m=0}^{n} \frac{f^m(v)}{m!} \sum_{v' \in \mathrm{prev}(v)} \frac{\alpha_{n-m}(v')}{(n-m)!} \quad (v \neq \mathrm{src}) \tag{32}$$

すなわち，$v \neq \mathrm{src}$ の場合，数列 $\left\{\frac{\alpha_n(v)}{n!}\right\}$ は，2 つの数列 $\left\{\frac{f^n(v)}{n!}\right\}$ と $\left\{\frac{\alpha_n(v')}{n!}\right\}$ との畳み込み演算，および添え字 n に無関係な線型演算によって計算できる．

ここで離散フーリエ変換に対する畳み込み定理の変種について言及しておこう．$N \in \mathbb{N}_0$ が与えられているとする．$2N$ 個の数からなり，後ろ半分の N 個が 0 埋めされた 2 つの数列 $\{a_0, \ldots, a_{N-1}, 0, \ldots, 0\}$ と $\{b_0, \ldots, b_{N-1}, 0, \ldots, 0\}$ に対して，数列 $\{c_n\}$ が以下の畳み込みで定義されているとする．

$$c_n := \sum_{m=0}^{n} a_m b_{n-m} \tag{33}$$

このとき，$\{c_n\}$ の離散フーリエ変換が，$\{a_n\}$ と $\{b_n\}$ の各々の離散フーリエ変換によって得られる 2 つの数列の要素毎の積として計算される．すなわち，

$$\mathcal{F}(\{c_n\}) = \mathcal{F}(\{a_n\}) \circ \mathcal{F}(\{b_n\}) \tag{34}$$

($\mathcal{F} : \mathbb{C}^{2N} \to \mathbb{C}^{2N}$ は離散フーリエ変換[⋆1]，$\circ$ は要素毎の積) である．このことに留意すると以下を得る．

定理．(フーリエ変換型一般化前向きアルゴリズム)

一般化前向きアルゴリズムの定理を言明した際に用いた表記を流用する．$N \in \mathbb{N}$ が与えられたとする．$^\forall v \in V, \quad n = 0, \ldots, 2N-1$ に対して，$\tilde{\alpha}_n(v)$ を以下のように $(V, \prec)$ の順に再帰的に定義する．

$$\tilde{\alpha}_n(\mathrm{src}) := \tilde{f}_n(\mathrm{src}),$$
$$\tilde{\alpha}_n(v) := \phi(v)\tilde{f}_n(v) \sum_{v' \in \mathrm{prev}(v)} \tilde{\alpha}_n(v') \quad (v \neq \mathrm{src}) \tag{35}$$

ただし，

$$\left\{\tilde{f}_n(v)\right\} := \mathcal{F}\left(\left\{\frac{f^0(v)}{0!}, \ldots, \frac{f^{N-1}(v)}{(N-1)!}, \underbrace{0, \ldots, 0}_{N\text{ 個}}\right\}\right) \tag{36}$$

である．このとき，

$$\mathcal{F}\left(\left\{\frac{\alpha_0(v)}{0!}, \ldots, \frac{\alpha_{N-1}(v)}{(N-1)!}, \underbrace{0, \ldots, 0}_{N\text{ 個}}\right\}\right) = \left\{\tilde{\alpha}_0(v), \ldots, \tilde{\alpha}_{N-1}(v), \underbrace{0, \ldots, 0}_{N\text{ 個}}\right\} \tag{37}$$

が成り立つ．

証 明 .

(sketch) α の定義 (23) の両辺をフーリエ変換した上で，フーリエ変換の線型性，およびフーリエ変換に対する畳み込み定理 (34) を利用することで得られる．□

上記の定理は，(35) に基づいた前向きの再帰計算で $\tilde{\alpha}_n(v)$ を計算すれば，それを離散逆フーリエ変換することによって $\alpha_n(v) \quad (n = 0, \ldots, N-1)$ が得られることを意味する．

同様に，一般化前向き後ろ向きアルゴリズムに対してもそのフーリエ変換に対応した版が得られる．

定理．(フーリエ変換型一般化前向き後ろ向きアルゴリズム)

一般化前向き後ろ向きアルゴリズムの定理，およびフーリエ変換型一般化前向きアルゴリズムの定理を言明した際に用いた表記を流用する．$^\forall v \in V, \quad n = 0, \ldots, 2N-1$ に対して，$\tilde{\beta}_n(v)$ を以下のように $(V, \prec)$ の逆順に再帰的に定義する．

$$\tilde{\beta}_n(\mathrm{snk}) := 1,$$
$$\tilde{\beta}_n(v) := \sum_{v' \in \mathrm{next}(v)} \phi(v')\tilde{f}_n(v')\tilde{\beta}_n(v') \quad (v \neq \mathrm{snk}) \tag{38}$$

このとき，

$$S_n := \frac{1}{n!} \sum_{\pi \in \Psi(G)} \Phi(\pi) F^n(\pi) \acute{F}(\pi)$$
$$\tilde{S}_n := \sum_{v \in V} \acute{f}(v)\tilde{\alpha}_n(v)\tilde{\beta}_n(v) \tag{39}$$

として以下が成り立つ．

$$\mathcal{F}\left(\left\{S_0, \ldots, S_{N-1}, \underbrace{0, \ldots, 0}_{N\text{ 個}}\right\}\right) = \left\{\tilde{S}_0, \ldots, \tilde{S}_{N-1}, \underbrace{0, \ldots, 0}_{N\text{ 個}}\right\} \tag{40}$$

証 明 .

(sketch) β の定義 (28) の両辺をフーリエ変換し，また (27) の両辺をフーリエ変換したものに畳み込み定理 (34) を適用することで得られる．□

3.3 計 算 量

$(n_1, \ldots, n_K)$-次の前向きアルゴリズムを実行するためには $(|V|+|E|)(n_1+1)^2 \cdots (n_K+1)^2$ に比例した時間計算量が必要となる．また，$(n_1, \ldots, n_K)$-次の前向き後ろ向きアルゴリズムを実行するためには，α および β を計算するための各々 $(|V|+|E|)(n_1+1)^2 \cdots (n_K+1)^2$ に比例する時間計算量，計算した α および β を記録するための $(|V|+|E|)(n_1+1)\cdots(n_K+1)$ に比例した空間計算量，および (27) を計算するための $(|V|+|E|)(n_1+1)^2 \cdots (n_K+1)^2$ に比例した時間計算量が必要となる．

⋆1 離散フーリエ変換の規格化定数として 1 を，離散逆フーリエ変換の規格化定数として $\frac{1}{2N}$ を取るものとする．

3.2 節で述べたフーリエ変換の形式で計算する場合，高速フーリエ変換を用いることにより高速化が可能となる．$(n_1,\ldots,n_K)$-次の一般化前向きアルゴリズムは $(|V|+|E|)(n_1+1)\cdots(n_K+1)\log(n_1+\cdots+n_K+K)$ に比例する時間計算量を要する．また，$(n_1,\ldots,n_K)$-次の一般化前向き後ろ向きアルゴリズムは $(|V|+|E|)(n_1+1)\cdots(n_K+1)\log(n_1+\cdots+n_K+K)$ に比例する時間計算量と $(|V|+|E|)(n_1+1)\cdots(n_K+1)$ に比例する空間計算量を要する．

4. Forward-Backward アルゴリズムの一般化による切り詰めたテイラー級数展開

$F(\pi)$ の任意の自然数べきを $\pi\in\Psi(G)$ に関して総和計算することが，3 節で述べたアルゴリズムにより効率的に可能であることが分かった．しかしながら，2 節で言及したように，条件 (4) を満たす関数 $F(\mathbf{y})$ に対して非線型関数を適用したものを可能なすべての出力候補系列に関して総和したいという要求もある．このような場合でも，対象の非線型関数がある一定の条件を満たせば，次の事実から効率的でかつ厳格な近似計算が可能となることが分かる．

注 記 ．

複素数値関数 $h(x)$ が $x_0\in\mathbb{C}$ を中心とする収束半径 $r\geq 0\quad(r\in\mathbb{R})$ のテイラー展開

$$h(x)=\sum_{m=0}^{\infty}a_m(x-x_0)^m\qquad(a_m\in\mathbb{C})\tag{41}$$

を持つとする．このとき，(4) を満たす $\Psi(G)$ 上の関数 $F(\pi)$ に関して，$^\forall\pi\in\Psi(G)$ に対して $|F(\pi)-x_0|<r$ ならば

$$\begin{aligned}\sum_{\pi\in\Psi(G)}h(F(\pi))&=\sum_{\pi\in\Psi(G)}\sum_{m=0}^{\infty}a_m(F(\pi)-x_0)^m\\&=\sum_{m=0}^{\infty}a_m\sum_{\pi\in\Psi(G)}(F(\pi)-x_0)^m\end{aligned}\tag{42}$$

である．(42) 中の式変形における最後の和の交換，$\sum_{\pi\in\Psi(G)}\sum_{m=0}^{\infty}(\cdot)=\sum_{m=0}^{\infty}\sum_{\pi\in\Psi(G)}(\cdot)$ が可能であることはテイラー展開の一様収束性から直ちに保証される．

ここでたとえば

$$\begin{aligned}F'(\pi)&:=\sum_{v\in\pi}f'(v)\\f'(v)&:=\begin{cases}f(v)-x_0 & v=\mathrm{src},\\ f(v) & \text{otherwise}\end{cases}\end{aligned}\tag{43}$$

とでも定義しなおせば，(42) の各項

$$\sum_{\pi\in\Psi(G)}(F(\pi)-x_0)^m=\sum_{\pi\in\Psi(G)}F'^m(\pi)\quad(m\in\mathbb{N}_0)\tag{44}$$

は先に述べた一般化前向きアルゴリズムで効率的に計算可能な形式であることが分かる．

ここでテイラー展開の高次項を適当に打ち切れば $\sum_{\pi\in\Psi(G)}h(F(\pi))$ を近似計算することができる．この近似は厳密である．厳密である，という言葉の意味はすなわち，

(1) 打ち切りの次数を上げることで希望する任意の近似精度を得ることが可能である．これはテイラー展開の収束性から直ちに保証される．

(2) 打ち切ったときの近似誤差の上限を得る解析的手法が存在する．すなわち，その近似誤差の上限はテイラー公式の剰余項の上限として得られる．

を意味する．ただし，ここで言及している誤差とは計算の打ち切り誤差のことだけであり，実際の計算機上で計算する際に生じるその他の誤差，すなわち丸め誤差・桁落ち・情報落ちなどについてはこの議論の枠外であることに注意されたい．結局，h が上記の条件を満たせば $\sum_{\pi\in\Psi(G)}h(F(\pi))$ は効率的かつ厳密に近似することが可能であるといえる．

5. 実　　験

確率モデルを CRF とした際の期待 F 値 (ただし $\gamma=1$, すなわち F_1 値のみに限った) の計算を一般化前向きアルゴリズムで近似した際に，誤差と計算量がどう挙動するかに関して実験を行った．

計算対象である期待 F 値を再掲すると

$$\begin{aligned}E_{P(\mathbf{y}|\mathbf{x};\lambda)}[F_1(\mathbf{y})]&=\frac{1}{Z(\mathbf{x};\lambda)}\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})}\Phi(\mathbf{y})F_1(\mathbf{y})\\&=\frac{1}{Z(\mathbf{x};\lambda)}\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})}\Phi(\mathbf{y})\cdot\frac{2l(\mathbf{y})}{|\mathbf{y}|+|\hat{\mathbf{y}}|}\end{aligned}\tag{45}$$

である．ここで

$$\frac{2l(\mathbf{y})}{|\mathbf{y}|+|\hat{\mathbf{y}}|}=\frac{2l(\mathbf{y})}{x_0-(-|\hat{\mathbf{y}}|)}\sum_{m=0}^{\infty}\left(\frac{|\mathbf{y}|-x_0}{-|\hat{\mathbf{y}}|-x_0}\right)^m\tag{46}$$

というテイラー展開が成立することに注意されたい．ただし，x_0 は $^\forall\mathbf{y}\in\mathcal{Y}(\mathbf{x})$ に対して $||\mathbf{y}|-x_0|<||\hat{\mathbf{y}}|+x_0|$ となるように取るものとする．ここでは $x_0=(\min_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})}|\mathbf{y}|+\max_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})}|\mathbf{y}|)/2$ とした．

$$T_n:=\sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x})}\Phi(\mathbf{y})l(\mathbf{y})\left(\frac{|\mathbf{y}|-x_0}{-|\hat{\mathbf{y}}|-x_0}\right)^n\tag{47}$$

と置くと，

$$E_{P(\mathbf{y}|\mathbf{x};\lambda)}[F_1(\mathbf{y})]=\frac{2}{Z(\mathbf{x};\lambda)(x_0+\hat{\mathbf{y}})}\sum_{n=0}^{\infty}T_n\tag{48}$$

である．そして，T_n は $(n,1)$-次の一般化前向きアルゴリズムで計算できる形式の和である．

上記のテイラー展開を N 次項までで打ち切った上で各項の計算に必要な T_n を一般化前向きアルゴリズムで計算した値と，定義どおりに出力候補集合上で総和した値との相対誤差が，テイラー展開の打ち切り次数を上げていったときにどう挙動するかを，いくつかの異なる系列，異なるパラメタ設定において実験した．

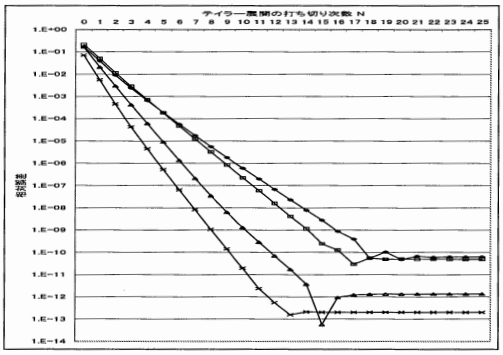

図 1 テイラー展開の打ち切り次数と相対誤差の関係

その結果を表したのが図 1 である．ただし，定義どおりの計算が可能なように $|\mathcal{Y}(\mathbf{x})| \sim 10^7$ 程度の比較的小規模な系列に限っている．計算は倍精度浮動少数 ($\varepsilon \sim 10^{-15}$) で行っている．片対数グラフである図 1 中で直線的に相対誤差が減少している，すなわち N に関して 1 次収束しているのは，T_n が幾何数列的に減少する事実を反映している．また，ある次数以上で相対誤差の減少が底打ちするのは打ち切り誤差以外の数値誤差によるものであり，このことを除けば，次数を上げることによって希望の精度を得られることが実証できている．もちろん，多倍長浮動少数を用いるなどすればさらなる近似精度を得られる．

また，紙面の都合上詳細な数値は省くが，フーリエ変換を用いない一般化前向き後ろ向きの実行に必要となる計算時間が，3.3 節で指摘したとおり，ほぼ $N^2|V|$ に比例することも実験から確認できた．

6. 結　　論

本稿では，前向き後ろ向きアルゴリズムを一般化し，従来の前向き後ろ向きアルゴリズムでは取り扱えなかった種類の計算も可能となることを示した．実験では，非線型な関数値を出力候補系列に対して総和する例として期待 F 値を例にとり，テイラー展開によって厳格に近似できることを実証した．

本稿で提案したアルゴリズムは，それ自身がそのまま有用となる類のものではないが，系列ラベリングの問題を解く際の極めて基礎的で汎用な数値計算ツールであることは間違いないであろう．系列ラベリングの問題を解く際に，出力候補集合上で何らかの総和を取るという計算は顕著な障害となることが多々あると考えられる．本稿で提案したアルゴリズムは，その顕著な障害を踏破するための基礎的な数値計算アルゴリズムとして大いに活用できることが期待できると考える．

今後は，本稿で提案したアルゴリズムを 2 節で例示した既存研究での手法と比較していきたい．また，本アルゴリズムを活用すれば，他の様々な確率モデル・パラメタ推定法・最適化手法が系列ラベリングの文脈に取り込めるものと期待しており，そのような可能性を積極的に模索していきたい．

参 考 文 献

1) Yasemin Altun, Alex J. Smola, and Thomas Hofmann. Exponential families for conditional random fields. In *Proc. of UAI 2004*, pp. 2–9, 2004.
2) A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Roy. Stat. Soc. Ser. B*, Vol.39, No.1, pp. 1–38, 1977.
3) Martin Jansche. Maximum expected f-measure training of logistic regression models. In *Proc. of HLT/EMNLP 2005*, pp. 692–699, October 2005.
4) Feng Jiao, Shaojun Wang, Chi-Hoon Lee, Russell Greiner, and Dale Schuurmans. Semi-supervised conditional random fields for improved sequence segmentation and labeling. In *Proc. of COLING-ACL 2006*, pp. 209–216, July 2006.
5) John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML 2001*, pp. 282–289, 2001.
6) John Lafferty, Xiaojin Zhu, and Yan Liu. Kernel conditional random fields: Representation and clique selection. In *in proc. of ICML 2004*, 2004.
7) GideonS. Mann and Andrew McCallum. Efficient computation of entropy gradient for semi-supervised conditional random fields. In *HLT-NAACL 2007; Companion Volume, Short Papers*, pp. 109–112, April 2007.
8) Sunita Sarawagi and WilliamW. Cohen. Semi-markov conditional random fields for information extraction. In *NIPS 17*, pp. 1185–1192, 2004.
9) S.V.N. Vishwanathan, NicolN. Schraudolph, MarkW. Schmidt, and KevinP. Murphy. Accelerated training of conditional random fields with stochastic gradient methods. In *Proc. of ICML 2006*, pp. 969–976, 2006.