添字付き計算グラフを用いた自動微分の実装

01307380 数理システム　　*田辺隆人 TANABE Takahito

1. 目的

n 個のベクトル変数と p 個のベクトル定数に依存している m 個のベクトル関数があり，その計算過程がやはりベクトルである中間変数 l 個を使って記述されているとする．関数，定数，中間変数，および変数（いずれもベクトル）を

$$x^k,\ k=1,\ldots,n+p+l+m \qquad (1)$$

と記述する．各計算過程はベクトル関数のコンポーネント同士の関係として

$$x^k_{W^k}=\sum_{C^k} f(x^{s_1}_{W^{s_1}},x^{s_2}_{W^{s_2}},\cdots,x^{s_r}_{W^{s_r}})$$
$$k=n+p+1,\ldots,n+p+l+m \qquad (2)$$

のように記述されているものとする．ここで，f は r 個の変数に依存する四則演算，初等関数の組み合わせで，$s^k_1,s^k_2,\cdots,s^k_r$ はベクトル x^k が依存している変数/定数/中間変数（ベクトル）の番号である．番号は変数ノードから依存関係の順に昇順に付加されているものとし，定数は変数の後に番号付けされているものとしている．すなわち（2）で，$k>s^k_1,s^k_2,\cdots,s^k_r$ であり，$x^1,x^2,\cdots,x^n$ は変数を示す．W^k は x^k の添字並びで，$x^k_{W^k}$ は添字付けにより x^k のコンポーネントを示している．C^k は和が取られる範囲を示しており，添字の集合である．その要素は $W^{s_1},W^{s_2},\cdots,W^{s_r}$ のいずれか少なくとも一つに含まれているとする．

大規模な数理モデルの大部分は繰り返し構造を持つので，変数や関数をベクトルとして表現すれば，そのコンポーネント同士にはすべて等質の関数関係が定義される．そのような場合には（2）の形の定式によってモデルを表現できる．行列と変数ベクトルの積に帰着する関係：

$$y_i=\sum_j A_{ij}x_j \qquad (3)$$

は（2）の最も簡単な例で，一般の大規模問題の記述に頻出する．この様な表現を多段にすることによって大規模で非線形なモデルも簡潔に表現することができる．例えば格付け推移確率行列推定問題は行列のN乗根をフィッティングによって求める問題になるが（3）に類する記述（行列積の積み上げ）で表現できる．多くのモデリング言語がこの繰り返し構造を利用して記述を簡略化しているが，本研究ではモデルを数値的に解析する際の自動微分アルゴリズムの実装にもこの構造を直接生かすことが可能であることを示す．

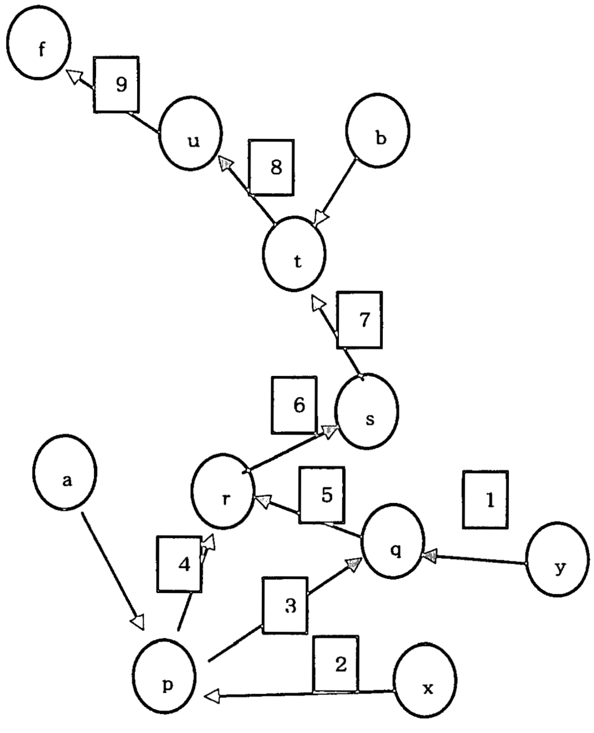
2. 要素導関数と一階微係数

例えば変数 x_j，y_{kl}，定数 a_{ij}，b_k によって記述される関数 f_k を計算するステップが，中間変数 $p_i,q_{ik},r_{ki},s_{ki},t_{ki},u_k$ を使って以下のように記述されているとする．

$$p_i=\sum a_{ij}\cdot x_j,\ q_{ik}=\sum p_i\cdot y_{kl},\ r_{ki}=p_i\cdot q_{ik}$$
$$s_{ki}=\exp(r_{ki}),\ t_{ki}=b_k{}^{L}s_{ki},\ u_k=\sum_i\sqrt{t_{ki}}$$
$$f_k=\log(u_k) \qquad (4)$$

このとき，例えば f_k の y_{kl} による偏微分係数 $\partial f_k/\partial y_{kl}$ は次のようにして求められる．

まず，（4）の記述が示す変数，関数，中間変数および定数同士の依存関係を次のようなグラフ（計算グラフと呼ばれる）の形で表現する．



ノードは変数・定数・中間変数・関数（ベクトル）のいずれかを示しており，辺は番号付けの若い方（変数側）を始点とするように向き付けられている．各辺には各辺の始点によって終点を微分した値（要素導関数，図では数字付きの四角）を対応付ける．例えば，「2」に相当する要素導関数は a_{ij}，「9」に相当する要素導関数は $1/u_k$，である．一

般に要素導関数は各ノードの値の四則演算あるいは初等関数の組み合わせとして記述され，この場合各ノードがベクトル量なので，要素導関数もベクトル量となる．続いて y_{kl} から f_k へのパス上に現れる要素導関数（「１」，「５」，「６」，「７」，「８」，「９」）の積を取ることによって $\partial f_k / \partial y_{kl}$ の表現：

$$\partial f_k / \partial y_{kl} = \sum_i p_i^{\,2} \cdot \exp(r_{ki}) \cdot b_k \cdot 1/2\sqrt{t_{ki}} u_k \quad (5)$$

が得られる．要素導関数の添字をそのまま引き継いでいるため，全体としてベクトル量の表現となっていることに注意されたい．ノード q_{ik}，u_k には集積演算が含まれているので i による集積演算が必要になっている．集積演算を行う添字（ここでは i）はパス上にあるノードの持つ添字に関する集合操作により機械的に決定することができる．

3. 一階微係数，二階微係数の表現

例えば数理計画アルゴリズムにおいてはすべての変数ノードからのすべての関数ノードの一階（二階）微係数をもれなく求める必要がある．そのためには計算グラフのノード上に

$$\underline{x}^i = \sum_{e\in\{(x^i \leftarrow y)\}} g_e \cdot \underline{y}, \qquad x^i \in \{\text{中間変数, 関数}\}$$

$$\underline{x}^i = x^i, \quad x^i \in \{\text{変数}\}$$

$$\underline{x}^i = 0, \quad x^i \in \{\text{定数}\} \qquad (6)$$

として定められる式を定義する．ただし $\{(x^i \leftarrow y)\}$ はノード x^i を終点とするすべての辺の集合，g_e は辺 e 上にある要素導関数を示すものとする．(6) の関係からすべての $\underline{x}^i$ を $x^i \in$ {変数} と g_e の積[1]のみを含む式に変形することができる．同様に

$$\overline{x}^i = \sum_{e\in\{(x^i \rightarrow y)\}} g_e \cdot \overline{y}, \quad x^i \in \{\text{変数, 中間変数}\}$$

$$\overline{x}^i = x^i, \quad x^i \in \{\text{関数}\}$$

$$\overline{x}^i = 0, \quad x^i \in \{\text{定数}\} \qquad (7)$$

を定義することができる．この関係からすべての $\overline{x}^i$ は $x^i \in$ {関数} と g_e の積のみを含む式に変形することができる．

変形後の $\underline{x}^i$，$\overline{x}^i$ について，$\overline{x}^i \cdot \underline{x}^i$ を計算すると，関数ノードと変数ノードの積（例えば（$x \cdot f$））が表れる．その係数に対応する要素導関数 g_e の積は $\partial f / \partial x$ のベクトル表示となっている．さらに，計算グラフの最小のノードセパレータ集合を S とし，$x^i \in S$ であるすべての x^i において，$\overline{x}^i \cdot \underline{x}^i$ の計算によって得られる微係数の表現を集めれば，すべての変数ノードによるすべての関数ノードの一階微係数の表現を求めることができる．

二階微係数を求めるには次の式を計算する．

$$\overline{x}^i \cdot \sum_{e_1,e_2 \in\{(x \leftarrow y)\}} h_{e_1,e_2} \cdot \underline{y}_1 \cdot \underline{y}_2 \qquad (8)$$

ここで $\underline{y}_1, \underline{y}_2$ は e_1, e_2 の始点に対応するノード，h_{e_1,e_2} は x^i の $\underline{y}_1, \underline{y}_2$ による二階微係数（二階の要素導関数）の値である．

（８）には関数ノードと変数ノード，変数ノード二つの積（例えば $x_1 \cdot x_2 \cdot f$）が表れるが，この係数である g_e の積が $\partial^2 f / \partial x_1 \partial x_2$ となっている．計算グラフ中，零でない二階の要素導関数を持つすべてのノードにおいて（８）を集めれば，すべての変数ノードによるすべての関数ノードの二階微係数のベクトル表現を得ることができる．

４．効率化について

（６），（７）によって定義される $\underline{x}^i$，$\overline{x}^i$ を変形してそれぞれ変数と関数のみを含む式にする具体的な方法（（６），（７）右辺に表れる $\underline{x}^i$，$\overline{x}^i$ の消去順序）は一意ではない．要素導関数がスカラである通常の場合にも，その選択が演算回数に影響することが知られており，消去順序決定のヒューリスティクスが提案されている[2, Chap8]．要素導関数がベクトルになる場合には，消去順序の変更は要素導関数ベクトル同士を行列として掛け合わせる順序に影響するため，その効果はさらに顕著に現れる．したがって要素導関数ベクトルのサイズをあわせて考慮する消去順序を工夫することにより，計算の効率化の余地がある．

謝辞:

本研究について貴重なご助言をいただきました中央大学の久保田光一教授にこの場を借りまして御礼申し上げます．

参考文献

[1] 久保田光一，伊理正夫，アルゴリズムの自動微分と応用，コロナ社，1998

[2] Andreas Griewank, Evaluating Derivatives, SIAM, 2000.

[1] ここで各 g_e の積は前章で述べた集積演算操作を含むものとする．