

高速自動微分法のためのプリプロセッサ
—実際の問題への応用例と問題点—

東京大学工学部計数工学科 久保田光一 (Koichi Kubota)

1. はじめに

高速自動微分法は変数の数が多い大規模の問題になるほど高速性を発揮する手法である。しかし、使用する計算機の資源（特に記憶容量）によって扱うことのできる問題の規模が制限される。本稿では、比較的大規模かつ実際的な問題に高速自動微分法を適用して、高速自動微分法の実際的な効力および問題点などを報告する。ここでは、我々が開発した高速自動微分法の利用のための FORTRAN プリプロセッサ PADRE2 [3]を用いて実験を行った。

ここで取り上げた比較的大規模かつ実際的な問題というのは、

- (a) 分子設計における、分子エネルギーの最小化問題（共役勾配法で解く際にヘッセ行列とベクトルとの積が必要）、
- (b) 経済モデルの感度解析（感度を表す偏導関数が必要）

の2種である。これらの問題を解くために、

- (IP-a) 分子エネルギーを計算するプログラム、
- (IP-b) 経済モデルをシミュレートするプログラム

によって関数を記述した。これらを PADRE2 の入力として、

- (OP-a) 分子エネルギーの値とその偏導関数を計算するプログラム、
- (OP-b) 経済モデルの関数値とその偏導関数を計算するプログラム

を作成した。（なお、IP-a, IP-b の両方ともその中に条件分岐, DO文, GOTO 文等を含んでいる。）これらの翻訳・実行の経験に基づいて PADRE2 を利用するのに必要な作業（オーバーヘッド）を示し、IP の実行時間・領域と OP の実行時間・領域の比較を行い、PADRE2の性能を評価し、問題点を挙げ、その改良法を示す。高速自動微分法については、文献 [1,2,3] に譲る。

なお、実験で使用した計算機は、東京大学大型計算機センターの副システム VAX-8600, OSは ULTRIX, 言語は f77 である（主記憶20Mbyte, 4.5MIPS）。ここで行った計算機実験は、使用する計算機に依存しており、必ずしも一般的な結果を与えるとは限らないが、高速自動微分法を実際に使用する上での指針となると考える。

2. PADRE2 の概要

PADRE2 は、FORTRAN 副プログラム IP (Input Program) を読み込み、FORTRAN 副プログラム OP (Output Program) を生成する FORTRAN プリプロセッサであり、PADRE [4]の改良版である。OP は、高速自動微分法に基づいて、IP の形で記述された関数の勾配、ヤコビ行列とベクトルの積、ヘッセ行列とベクトルの積などの値を計算する副プログラムである（IP を PADRE2 に入力する時にどの値を計算するかを指定される）。PADRE2 はプログラミング言語Cで記述（約6000行）されており、マイクロコンピュータ PC-9801 でも動作する（ここでの応用例の OP は PC-9801 でも生成することができる）。

3. 問題の規模と実験結果

問題の規模は、①プログラムの規模、②計算の規模の2種に分けて考えることができる。ここでは、プログラムの規模をプログラムの行数（コメントを除く）で測る。また、プログラムが終了する（関数値を計算する）までに遂行される演算の回数で計算の規模を測る。もちろん、計算を実行する時に入力されるデータによって計算の規模は変化する。PADRE2を利用する場合には、①はOPの行数に関係し、②はOPを実行する場合に必要な記憶領域に関係する。例えば、20元の線形方程式系をLU分解で求解する場合は、プログラムの規模は小さい（20行ぐらい）が、計算の規模は大きい（演算回数は5000以上）と考える。

3. 1. 実験で取り上げた問題

3. 1. 1. 応用例1（スカラー値関数）

IP-aは、原子の結合角を変数として、この角度から分子のエネルギーを計算するプログラムである（もとは24個のサブルーチンを使用して記述していたものを展開して1個の副プログラムにした）。OP-aは、分子のエネルギーの値、結合角に関する勾配およびヘッセ行列とベクトルとの積のそれぞれの値を計算するプログラムである。実質4変数になる簡単なデータを入力した。

3. 1. 2. 応用例2（ベクトル値関数）

IP-bは、“外生変数”を変数、“内生変数”を関数値とする経済モデル（55変数99関数のベクトル値関数）をシミュレートするプログラムである。OP-bは、関数値に加えてヤコビ行列とベクトルとの積を計算するプログラムである。

3. 2. プログラムの規模に関して

ここでは、プログラムの規模に関してIPとOPを比較する。

3. 2. 1. PADRE2を利用することによるオーバーヘッド

IPからOPを作り、OPを翻訳して実行できる状態にするまでの手間が、PADRE2を利用することによるオーバーヘッドである。具体的には、

- (1) PADRE2が受理できるように、いくつかの制約を満たすようなIPを作成すること
- (2) OPに合ったメインルーチンを用意し、これとOPとを翻訳して実行できる状態にすること

である。(1)のオーバーヘッドは、PADRE2のIPに対する次の制約：

- ・IPの中ではユーザー定義の副プログラムを使用できない（それらはすべて展開しておかなければならない）、
- ・DATA文は使用できない（DATA文を代入文に変換しなければならない）、
- ・DO文の末端文をCONTINUE文に限る

に起因する。IPを新しく作成する場合は、これらの制約を満たすように作成すれば良いが、既存のプログラムを変更してIPとする場合には、この変更は現在のところ人間が行うので、誤りやすい。（IP-aでは、DATA文の代わりに、約300行の代入文を追加しなければならなかった。）

IPの大きさ、IPを翻訳するための時間、IPのオブジェクトの大きさ、実行モジュールの大きさなどを表1に示す。

OPの諸元を表2に示す。OP-aはIP-aの7倍の行数になり、OP-bはIP-bの29倍の行数になった。現PADRE2は、IPにおける一つの演算あたりOPの中に7行の実行文を生成する。OP-bの方が行数増加の割合が大きくなったのは、IP-bにおいて一行に多くの演算を（つまり、比較的複雑な式を）使用しているからである。翻訳時間はOP-aは7倍、OP-bは13倍となり、ほぼプログラムの文字数に比例して時間がかかった。

表1. IP の諸元

	(a)分子エネルギー	(b)経済モデル
行数 ^{*1}	2404 行	347 行
文字数 ^{*2}	60 Kbyte	16 Kbyte
翻訳時間 ^{*3}	64 秒	17 秒
オブジェクト の大きさ ^{*4}	82 Kbyte	24 Kbyte
実行時間 (プリプロセス時間 ^{*5})	0.73 秒 24 秒	3.7 秒 9 秒)

- *1 コメント行を含めない
- *2 プログラムの大きさをバイトで測ったもの
- *3 ULTRIX の f77 コマンドでコンパイルに要した時間
- *4 シンボルテーブル等のストリップをしてない
- *5 PADRE2 で OP を生成するのに必要な時間

表2. OP の諸元

	(a)分子エネルギー	(b)経済モデル ^{*1}	
		[row]	[col]
行数	17189 行	9900	9879 行
文字数	420 Kbyte	246	245 Kbyte
翻訳時間	395 秒	221	218 秒
オブジェクト の大きさ	487 Kbyte	280	279 Kbyte

- *1 ベクトル値関数に対しては、OP は [row]ヤコビ行列に左からベクトルを掛けるもの、および [col]右から列ベクトルを掛けるもの の2種類ある。

3.3. 計算の規模に関して

ここでは、実行時間および実行に必要な領域について IP と OP を比較する。高速自動微分法は、計算グラフという形で計算の履歴を保持する。このために、計算の規模に比例した領域を OP の実行時に必要とする。現在の計算機資源で、どのくらいの計算の規模の問題にまで高速自動微分法を適用できるかを調べることは大切である。

OP の実行時間のうち、関数計算に必要な時間、勾配等計算に必要な時間を分けて表3に示す。OP は関数計算と同時に計算グラフを動的に作成しているので関数の計算部分だけをとっても IP の計算時間より時間がかかる。関数計算と偏導関数計算と合わせて OP-a は IP-a の10倍程度、OP-b は IP-a の4倍程度であることが観察された。これにより、

- ①変数の数が10個以上あれば、数値微分で勾配を計算するよりも速く、勾配

の正確な値とヘッセ行列とベクトルとの積を得られる、
 ②数値微分の4倍の時間で偏導関数（ヤコビ行列とベクトルとの積）の正確な値を得られる
 ということが言える。[1,3]で行われている高速自動微分法の手間の評価は計算グラフ上を操作する手間を無視しているが、ここで取り上げたような関数の計算時に動的に計算グラフの構造が定まっていくものに対しても高速自動微分法は十分高速性を発揮していると言えよう。

ここで入力したテストデータに関して、関数の計算グラフを保持するためにOPにおいて余分に必要となる領域を表4に示す。OP-a, OP-bともに1Mbyte以下の領域があればよいので、現在のいわゆるワークステーション級以上の計算機であれば、必要な記憶領域を十分賄うことができる。計算グラフ自身を記憶するための領域の他に余分に必要な領域は、計算グラフを作成するための名前表のためのもので、これは問題(a), (b)ともに約80Kbyteであった。(この名前表は、プログラム中で使用している変数名の数と配列の大きさに比例し、プログラムの規模に関係する。)

表3. OPの実行時間

	(a)分子エネルギー	(b)経済モデル	
		[row] ¹	[col] ²
関数計算と計算グラフ作成のための時間	2.7 秒	9.2	9.2 秒
勾配計算等のための時間	4.7 秒	4.2	2.4 秒

¹ ヤコビ行列に左から行ベクトルを掛ける

² ヤコビ行列に右から列ベクトルを掛ける

表4. OPの実行時に必要な記憶領域

	(a)分子エネルギー	(b)経済モデル
	計算グラフの頂点数 大きさ	約 36000 頂点 870 Kbyte
1 頂点あたりの 整数データ	3 個	3 個
実数データ (バイト数)	3 個 24 byte	2 個 20 byte)
名前表の大きさ	約 80 Kbyte	約 80 Kbyte

4. PADRE2 を利用することによる問題点と改良法

4. 1. 問題点

(問題点 i) OP のプログラムの規模が大きすぎること。

IP-a のプログラムの規模が大きいために、OP-a は1個の巨大な副プログラムになった。幸いなことに、今回使用した ULTRIX の f77 は、OP-a を翻訳し、実行できたので、表2の結果が得られた。しかし、汎用大型計算機 M680 のコンパイラは、OP-aが1個の副プログラムとしては大きすぎるため、これを翻訳できなかった。(OPが単一の副プログラムでなく複数の副プログラムであれば、全体の行数が少々増加しても、このような問題を回避できる(複数の副プログラムからなる数万行のプログラムは珍しくない)。)

(問題点 ii) 計算の規模が大きく、OP を実行するための記憶領域が不足する
場合があること。

高速自動微分法自身が関数の計算時間に比例する記憶領域を必要とし、一般的にはこの問題点を回避できない(しかし、関数の構造を個別に解析して、記憶領域を減らすことができる場合もある[5])。

4. 2. 改良法と改良後の実験結果

問題点(i)については、OP の中で計算グラフの作成のための副プログラム(これはユーザーが自由に定義することは出来ない)を使用して OP の行数を減少させるという改良方法がある。すなわち、現在の PADRE2 が OP の実行速度を重視して1演算あたり6行(演算自身に必要な1行を加えると7行)費やして計算グラフを作成していたものを、1行の“副プログラム呼び出し”に変更して行数を減らす方法である。もちろん、こうして作成した OP の実行時間は増加してしまうが、OP の実行時の速度よりも OP の行数が少ない事を優先し、行数の少ないプログラム OP-small をも出力できるように PADRE2 を変更した。これを用いて、OP-a-small, OP-b-small を作り出し、IP-a, IP-b に対する比較を行った結果を表5に示す。OP-a-small は約9100行になり、OP-b は約4700行になり、それぞれ

表5. OP-small の諸元

	(a)分子エネルギー		(b)経済モデル	
			[row] ^{*1}	[col] ^{*2}
行数	9121 行		4715	4694 行
文字数	278 Kbyte		155	155 Kbyte
翻訳時間	224 秒		109	110 秒
オブジェクトの大きさ	357 Kbyte		200	199 Kbyte
関数計算と計算グラフ作成のための時間	2.9 秒		10.0	10.0 秒
勾配計算等のための時間	4.7 秒		4.1	2.4 秒

*1 ヤコビ行列に左から行ベクトルを掛ける

*2 ヤコビ行列に右から列ベクトルを掛ける

OP-a, OP-b の約半分になった。翻訳時間も OP-a-small は CPU 約225秒, OP-b-small は CPU 約110秒に減少した。OP-a の実行時間に対して OP-a-small の関数計算と計算グラフ作成のための時間は8%増加, OP-b に関しては9%増加しただけであった。このことから, 1割程度の実行時間の増加を許すならば, OP の行数を減少させる方法は翻訳時間などの減少に有効であると言える。ただし, 実行時に必要な記憶領域は変わらない。(M680 のコンパイラは, OP-a-small の翻訳を終了するが, これを実行すると, エラーを起こす。最終的には, OP-a-small を3個の副プログラムに分割することによって, M680 でも翻訳・実行できるようになった。)

5. おわりに

ここでとりあげたような比較的大規模な関数についても, 高速自動微分法を利用できることを実証した。しかし, 計算機によっては, プログラムの規模が大きすぎて翻訳すらできないという問題点が発生した。これは当初予想していなかったことである(幸い, ULTRIX の f77 では問題無かった)。この直接の原因は, 元々のプログラム(IPのもととなったプログラム)でのサブルーチンをインライン展開したことによる。このような問題点を避けてさらに使い易くするために, IP 中での利用者定義の副プログラムの使用を許可するように PADRE2を改良していくつもりである。

現在 PADRE2 は, VAX-8600, SUN, FACOM および PC9801 で動作している。マイクロコンピュータ PC-9801 の上では, IP から OP を生成することは可能だが, IP が十分小さくない限りその OP を翻訳・実行させることはできない(記憶領域不足が原因)。しかし, そういった小規模の IP であっても, ここでは詳しく述べなかったが, 実験的に丸め誤差解析等を行うためには有用であろう。

謝辞

実例にあげた問題を提供して下さった東レシステムセンターの橋本茂司氏, 竹内正道氏, 寺前正一氏, 電力中央研究所の松井正一氏に感謝致します。また, 高速自動微分法の研究はもちろん, それ以外にも多大な援助をしてくださった東京大学工学部伊理正夫教授, 東京大学工学部室田一雄助教授に感謝致します。

参考文献

- [1] M. Iri (1984): Simultaneous Computation of Functions, Partial Derivatives and Estimates of Rounding Errors — Complexity and Practicality. Japan Journal of Applied Mathematics, Vol. 1, No. 2, pp. 223-252.
- [2] 伊理 正夫, 土谷 隆, 星 守 (1985): 偏導関数計算と丸め誤差推定の自動化の大規模非線形方程式系への応用. 情報処理, Vol. 26, No. 11, pp. 1411-1420.
- [3] 伊理 正夫, 久保田 光一 (1986): 高速微分法とその応用. 第7回数理計画シンポジウム論文集, pp. 159-184.
- [4] 岩田 憲和 (1984): 偏導関数計算の自動化. 東京大学大学院工学系研究科情報工学専門課程修士論文.
- [5] 戸川 隼人 (1987): 伊理のアルゴリズムの線形計算向けチューニング. 応用数学合同シンポジウム研究報告集(於京都大学数理解析研究所 昭和62年12月21~23日), pp. 1-2