自動微分の資料

B4　望月陽介

June 1, 2021

Arilim Güneß Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, Jeffrey Mark Siskind.
"Automatic Differentiation in Machine Learning: a Survey",
Journal of Machine Learning Research 18 (2018) 1-43

1　モチベーション

解析解が求められない時のパラメータ推定は、目的関数の微分値を求め、その勾配方向にパラメータを変化させることで目的関数を最大化または最小化する。収束条件のために正確な微分値を求めることだけでなく、従来のモデルよりも何倍も大きいパラメータ数を扱うニューラルネットのようなモデルでは特に、計算速度を上げることが必要となる。ここでは誤差逆伝播法のように一般的に使われるようになっている、自動微分について他の微分法に対する長所とともに紹介する。

2　自動微分以外の微分法

2.1　数値微分 (Numerical)

数値微分は、元関数のある地点での値を用いることによって、微分の近似値を求める方法である。最も簡単な形として、微分の定義に基づく前進差分が用いられる。関数 $f : \mathbb{R}^n \to \mathbb{R}$ に対して勾配 $\nabla f = (\frac{\partial f}{\partial \boldsymbol{x}_1}, \ldots, \frac{\partial f}{\partial x_n})$ は

$$\frac{\partial f(\boldsymbol{x})}{\partial x_i} \approx \frac{f(\boldsymbol{x} + h\boldsymbol{e}_i) - f(\boldsymbol{x})}{h} \tag{1}$$

として与えられる。$\boldsymbol{e}$ は i 次元方向の単位ベクトル、$h > 0$ は微小のステップサイズである。

数値微分は多くの場合において解が不安定であることが知られている。この不安定性は h が大きさを持つことに起因する打ち切り誤差と、コンピュータの演算内で行われる下位桁数の切り捨てに起因する丸め誤差による。前者はステップサイズ h を小さくすることでゼロに近づくが、その場合丸め誤差は大きくなる傾向にある。

また、計算精度が改善された場合でも、高次元の関数 $f : \mathbb{R}^n \to \mathbb{R}^m$ のヤコビ行列を求める場合、$2mn$ 回の計算を行う必要があり、計算精度と性能のトレードオフに直面するという問題がある。

2.2　数式微分 (Symbolic)

数式微分は、以下 (2) に記す微分の法則を適応することによって微分表現を得る自動的な操作のことを指す。数値微分と違い、微分表現を出力することのできる数式微分では、解析解を求めることのできる場合もある。その一方で、微分表現が元の関数よりも指数関数的に大きい長さで記述されるため、計算の実行は決して効率的とは言えない (Table 1)。

関数 $h(x) = f(x)g(x)$ を考える。(2) の法則を適応すると、$h(x)$ と $\frac{d}{dx}h(x)$ は共通の要素 $f(x)$ と $g(x)$ を持つことが分かる。さらに微分を進める際に、$f(x)$ と $\frac{d}{dx}f(x)$ の間にも共通する要素が現れる。これらを全て重複して計算する必要があるため、自動微分は指数関数的に大きい微分表現、それによる長い計算時間を要することになる。

$$\begin{cases} \dfrac{d}{dx}(f(x)+g(x)) & \rightsquigarrow \dfrac{d}{dx}f(x)+\dfrac{d}{dx}g(x) \\ \dfrac{d}{dx}(f(x)g(x)) & \rightsquigarrow \left(\dfrac{d}{dx}f(x)\right)g(x)+f(x)\left(\dfrac{d}{dx}g(x)\right) \end{cases} \tag{2}$$

Table 1: 数式微分による Expression Swell の例　関数 $l_{n+1}=4l_n(1-l_n), l_1=x$

n	l_n	$\frac{d}{dx}l_n$	$\frac{d}{dx}l_n$ (Simplified form)
1	x	1	1
2	$4x(1-x)$	$4(1-x)-4x$	$4-8x$
3	$16x(1-x)(1-2x)^2$	$16(1-x)(1-2x)^2-16x(1-2x)^2-64x(1-x)(1-2x)$	$16(1-10x+24x^2-16x^3)$
4	$16x(1-x)(1-2x)^2(1-8x-8x^2)^2$	$128x(1-x)(-8+16x)(1-2x)^2(1-8x+8x^2)+64(1-x)(1-2x)^2(1-8x+8x^2)^2-64x(1-2x)^2(1-8x+8x^2)^2-256x(1-x)(1-2x)(1-8x+8x^2)^2$	$64(1-42x+504x^2-2640x^3+7040x^4-9984x^5+7168x^6-2048x^7$

3　自動微分 (Automatic)

自動微分では、数値計算を、その微分が既に分かっている基礎的な計算操作有限個の組み合わせとして捉え、全体の計算を構築する各操作についての微分を、連鎖律に基づき計算していくことで、全体の微分値を求める。基礎的な計算操作とは普通、二項演算や単項演算、指数や対数、三角関数といった超越関数のことを指す。

どんな計算命令でも、入力変数、中間変数、出力変数から成る計算の軌跡を辿ることになる。このため、自動微分を用いることで、古典的な数学関数だけでなく、分岐や繰返しといった制御構造を利用したアルゴリズムの微分をも、この計算の軌跡を辿ることで求めることができる。このことは数式微分に比べた時の大きな優位性の一つである。

基礎的な計算の組み合わせとして関数を捉えた計算過程を、関数 $f:\mathbb{R}^2\to\mathbb{R}$、$y=f(x_1,x_2)=\ln x_1+x_1x_2-\sin x_2$ を例にして Table 2 と Table 3 の左側に示した。この時の入力変数、中間変数、出力変数は Figure 1 のように表すことができる。

Table 2: ボトムアップ自動微分の例、関数 $y=f(x_1,x_2)=\ln x_1+x_1x_2-\sin(x_2)$ の x_1 に対する微分を $(x_1,x_2)=(2,5)$ において求める。

Forward Primal Trace			Forward Tangent Trace		
v_{-1}	$=x_1$	$=2$	$\dot{v_{-1}}$	$=\dot{x_1}$	$=1$
v_0	$=x_2$	$=5$	$\dot{v_0}$	$=\dot{x_2}$	$=0$
v_1	$=\ln v_{-1}$	$=\ln 2$	$\dot{v_1}$	$=\dot{v_{-1}}/v_{-1}$	$=1/2$
v_2	$=v_{-1}\times v_0$	$=2\times 5$	$\dot{v_2}$	$=\dot{v_{-1}}\times v_0+\dot{v_0}\times v_{-1}$	$=1\times 5+0\times 2$
v_3	$=\sin v_0$	$=\sin 5$	$\dot{v_3}$	$=\dot{v_0}\times\cos v_0$	$=0\times\cos 5$
v_4	$=v_1+v_2$	$=0.693+10$	$\dot{v_4}$	$=\dot{v_1}+\dot{v_2}$	$=0.5+5$
v_5	$=v_4-v_3$	$=10.693+0.959$	$\dot{v_5}$	$=\dot{v_4}-\dot{v_3}$	$=5.5-0$
y	$=v_5$	$=11.652$	$\dot{y}$	$=\dot{v_5}$	$=5.5$

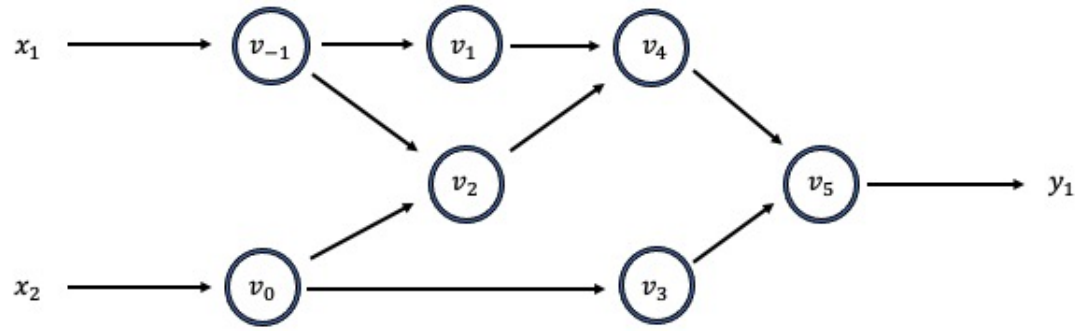

Figure 1: $y = f(x_1, x_2) = \ln x_1 + x_1 x_2 - \sin(x_2)$ の計算グラフ

3.1 ボトムアップ型

ボトムアップ型 (foward mode) の自動微分では、入力変数に対する中間変数の微分を、Figure 1 のような計算の軌跡を一つ一つ辿っていき、連鎖律に基づいて求めていく。これを出力変数の微分が求まるまで計算の軌跡を辿ることになる。Table 2 の右側では関数 $f : \mathbb{R}^2 \to \mathbb{R}$、$y = f(x_1, x_2) = \ln x_1 + x_1 x_2 - \sin x_2$ の x_1 に対する微分を求めているが、操作としては

$$\dot{v_i} = \frac{\partial v_i}{\partial x_1}$$

を連鎖律に基づき Figure 1 の計算の軌跡を辿りながら求めていく。

関数 $f : \mathbb{R}^n \to \mathbb{R}^m$ のヤコビ行列を求めることに拡大すると、ボトムアップ型は一回の操作で $\dot{\boldsymbol{x}} = \boldsymbol{e_i}$($\boldsymbol{e_i}$ は i 番目の単位ベクトル) に対する全ての出力変数の微分を求めることができる。つまり、$\boldsymbol{x} = \boldsymbol{a}$ において、一回の操作で求めているものは

$$\dot{y_j} = \left.\frac{\partial y_j}{\partial x_i}\right|_{\boldsymbol{x}=\boldsymbol{a}}, j = 1, \ldots, m$$

であり、$\boldsymbol{x} = \boldsymbol{a}$ でのヤコビ行列

$$\boldsymbol{J}_f = \left.\begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}\right|_{\boldsymbol{x}=\boldsymbol{a}}$$

の一列を得ることができる。よって、ボトムアップを n 回繰り返すことにより、ヤコビ行列は得られることが分かる。そのため、$\nabla \boldsymbol{f}$ を求めるための計算は、出力変数の個数 m が入力変数の個数 n より大きい場合には効率的に行える一方、n が m より大きい場合には効率的でないため、次に紹介するトップダウン型を用いることになる。

また、あるベクトルに沿った関数の微分値である、ヤコビ行列とベクトルの積

$$\boldsymbol{J}_f\boldsymbol{r} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}\Bigg|_{\boldsymbol{x}=\boldsymbol{a}} \begin{bmatrix} r_1 \\ \vdots \\ r_n \end{bmatrix}$$

は $\dot{\boldsymbol{x}} = \boldsymbol{r}$ と初期化してボトムアップの計算を行うことで、一度の操作で得ることができる点も自動微分の優れた点である。なお、ボトムアップ型の自動微分は二重数を用いて関数を計算することに落とし込むことができるが、ここでは割愛する。

3.2 トップダウン型

トップダウン型の自動微分では出力変数から微分値が後ろ向きに広がっていく形をとり、誤差逆伝播と同じ操作である。これは、出力変数 y_j の、v_i の変化に対する感度を表す

$$\bar{v_i} = \frac{\partial y_j}{\partial v_i}$$

を各計算に対して更新していくことで行う。

トップダウン型の自動微分は二つの段階を踏んで微分値を求めることになる。一つ目の操作は Table 3 の左側に相当し、中間変数と出力変数を求め、各変数間の計算グラフ上での依存性を記録する作業を行う。二つ目の操作は Table 3 の右側に相当し、各中間変数での出力変数の微分 $\bar{v_i}$ を、出力変数から入力変数へと後ろ向きに求めていく。関数 $y = f(x_1, x_2) = \ln x_1 + x_1 x_2 - \sin(x_2)$ の例に戻って考える。Table 3 の左側に示した基礎的な操作に対して、右側でその操作に対する微分律を用いた微分を行っている。出力変数の、各変数での微分 $\bar{v_i} = \frac{\partial y}{\partial v_i}$ を求めたい時、v_0 を例に取ると、Figure 1 から分かる通り、v_0 が y に影響を与えるのは v_2 と v_3 を通してである。そのため、微分値は、

$$\frac{\partial y}{\partial v_0} = \frac{\partial y}{\partial v_2}\frac{\partial v_2}{\partial v_0} + \frac{\partial y}{\partial v_3}\frac{\partial v_3}{\partial v_0} \qquad or \qquad \bar{v}_0 = \bar{v}_2\frac{\partial v_2}{\partial v_0} + \bar{v}_3\frac{\partial v_3}{\partial v_0}$$

の形で与えられる。Table 3 から分かるように、この微分値はトップダウン型自動微分では二つのステップ

$$\bar{v}_0 = \bar{v}_3\frac{\partial v_3}{\partial v_0} \qquad and \qquad \bar{v}_0 = \bar{v}_0 + \bar{v}_2\frac{\partial v_2}{\partial v_0}$$

によって求められる。一回トップダウン型を最後まで計算することで、出力変数の微分が全ての入力変数に対して求められることが分かる。

トップダウン型の利点は、入力変数が多い関数を扱う際に、各入力変数での出力変数の微分値を求める計算が非常に速い点にある。これは、$f : \mathbb{R}^n \to \mathbb{R}$ の場合には勾配 $\nabla f = \left(\frac{\partial y}{\partial x_1}, \ddots, \frac{\partial y}{\partial x_n}\right)$ を求めるために、ボトムアップ型は n 回の適用が必要であるのに対し、トップダウン型は一度の適用で求めることができることからも分かる。一般的に、関数 $f : \mathbb{R}^n \to \mathbb{R}^m$ について元の関数を計算する際の操作数を $ops(f)$ と表した時、$m \times n$ のヤコビ行列を計算するためにかかる時間は、ボトムアップ型では $n\,c\,ops(f)$、トップダウン型では $m\,c\,ops(f)$ と表せる。ここで c は $c < 6$ の定数で、一般的に $c \sim [2, 3]$ である。

また、ボトムアップ型と同様に、ヤコビ行列とベクトルの積

$$\boldsymbol{J}_f^{\boldsymbol{T}}\boldsymbol{r} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}\Bigg|_{\boldsymbol{x}=\boldsymbol{a}} \begin{bmatrix} r_1 \\ \vdots \\ r_m \end{bmatrix}$$

も、$\bar{\boldsymbol{y}} = \boldsymbol{r}$ と初期化してトップダウン型を適用することで求めることができる。

トップダウン型の自動微分は入力に対する微分計算の速さや、一度での中間変数に対する微分求まる点などの利点がある一方、関数の操作の数に比例して増加する容量の要求値という問題がある。これを解決する方法として、データフロー解析やチェックポイント戦略などが挙げられている。

Table 3: トップダウン自動微分の例、関数 $y = f(x_1, x_2) = \ln x_1 + x_1 x_2 - \sin(x_2)$ の各変数に対する微分を $(x_1, x_2) = (2, 5)$ において求める。

Forward Primal Trace			Reverse Adjoint Trace			
v_{-1}	$= x_1$	$= 2$	$\bar{x}_1$	$= \bar{v}_{-1}$		$= 5.5$
v_0	$= x_2$	$= 5$	$\bar{x}_2$	$= \bar{v}_0$		$= 1.716$
v_1	$= \ln v_{-1}$	$= \ln 2$	$\bar{v}_{-1}$	$= \bar{v}_{-1} + \bar{v}_1 \frac{\partial v_2}{\partial v_{-1}}$	$= \bar{v}_{-1} + \bar{v}_1 / v_{-1}$	$= 5.5$
v_2	$= v_{-1} \times v_0$	$= 2 \times 5$	$\bar{v}_0$	$= \bar{v}_0 + \bar{v}_2 \frac{\partial v_2}{\partial v_0}$	$= \bar{v}_0 + \bar{v}_2 \times v_{-1}$	$= 1.716$
			$\bar{v}_{-1}$	$= \bar{v}_2 \frac{\partial v_2}{\partial v_{-1}}$	$= \bar{v}_2 \times v_0$	$= 5$
v_3	$= \sin v_0$	$= \sin 5$	$\bar{v}_0$	$= \bar{v}_3 \frac{\partial v_3}{\partial v_0}$	$= \bar{v}_3 \times \cos v_0$	-0.284
v_4	$= v_1 + v_2$	$= 0.693 + 10$	$\bar{v}_2$	$= \bar{v}_2 \frac{\partial v_4}{\partial v_2}$	$= \bar{v}_4 \times 1$	$= 1$
			$\bar{v}_1$	$= \bar{v}_4 \frac{\partial v_4}{\partial v_1}$	$= \bar{v}_4 \times 1$	$= 1$
v_5	$= v_4 - v_3$	$= 10.693 + 0.959$	$\bar{v}_3$	$= \bar{v}_5 \frac{\partial v_5}{\partial v_3}$	$= \bar{v}_5 \times (-1)$	$= -1$
			$\bar{v}_4$	$= \bar{v}_5 \frac{\partial v_5}{\partial v_4}$	$= \bar{v}_5 \times 1$	$= 1$
y	$= v_5$	$= 11.652$	$\bar{v}_5$	$= \bar{y}$	$= 1$	