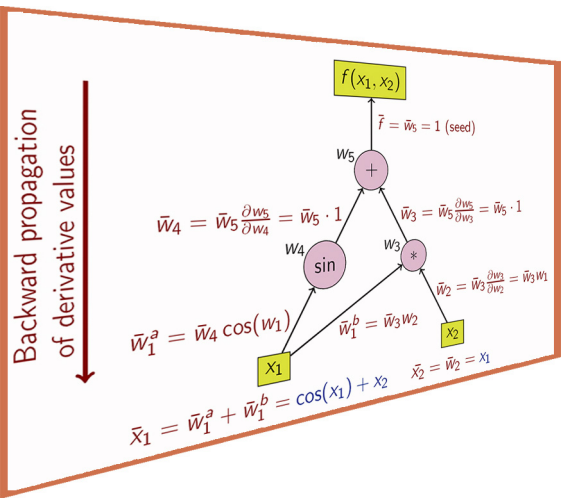

자동 미분의 공학문제 적용

Engineering Application of Automatic Differentiation

구 본 용
군산대학교 기계융합시스템공학부 부교수

1. 서론

최근 머신러닝 분야의 급속한 발전과 함께 인공지능 모델 최적화 즉 설계 변수에 대해 손실함수(loss function)를 최소화하는 컴퓨터 연산이 상상을 초월할 정도로 많아지고 있다. 이러한 인공지능 모델 학습에는 Quisi-Newton 알고리즘의 일종인 BFGS 또는 개선된 L-BFGS 알고리즘이 주로 사용되며 설계 변수(인공지능 모델에서는 가중치)에 대한 손실함수의 미분값 계산이 중요한 과정이 된다. 전통적으로 유한차분(finite differencing)에 기반하여 미분값을 구할 수 있으나 계산정확도와 시간이 오래 걸리는 단점이 있으며, 정확한 미분값 계산과 시간 단축을 위해 자동 미분(automatic differentiation, AD)이 인공지능 학습에 널리 사용되고 있다.

자동 미분은 기본적으로 미분값 계산과정을 체인룰로 바꾸어 계산하는 것이라고 할 수 있으며 컴퓨터 알고리즘으로 적용한 순방향 전파(forward propagation)의 경우 1964년 Wengert[1] 그리고 역방향 전파(backward propagation)의 경우 1976년 Linnainmaa[2]에 의해 제안되었다. 또한 국내에는 2000년 이후 관련 논문을 찾아 볼 수 있으며, 2004년 이재훈 등[3]이 전산유체해석에 자동 미분을 도입하여 공력 형상 최적설계를 수행한 논문이 대표적이다.

자동 미분을 공학문제의 최적화에 활용하기 위해서는 함수 계산 수준에서 소스코드를 수정하여야 하므로 인하우스 코드를 가지고 있는 연구그룹에서 활발하게 도입되었고, 또한 알고리즘 구현상의 이유로 C++, Fortran90과 같은 객체지향 언어를 사용하는 라이브러리가 다양하게 개발되어 왔다. 최근 인공지능 분야의 발전과 함께 Python, Julia, MATLAB과 같은 사용자 편의성이 높은 프로그래밍 언어에서도 자동 미분을 지원하고 있으며, 역행렬 계산 함수에도 자동 미분 알고리즘을 지원하고 있다.

공학 문제의 최적화에서 다수의 알고리즘은 설계 변수에 대한 목적 함수의 경사도를 계산하는 것이 필요하며 빠른 경사도 계산을 위해 해석적 설계민감도를 유도하는 것 또한 많은 연구가 이루어져 왔다. 이러한 해석적 방법은 엄밀한 결과를 제공함에도 불구하고 복잡한 수식유도와 보조방정식(adjoint equation)을 풀어야 하는 등 어려움이 존재한다. 따라서 여러 분야의 최적화 문제에서 해석적 설계민감도 계산을 대체하는 방법으로 자동 미분 알고리즘이 도입되고 있으며 본 원고에서는 자동 미분을 공학 문제에 적용한 최근의 연구를 소개하고자 한다.

2. 자동 미분 연산

이 장에서는 Wikipedia 자동 미분[4]에 기반하여 순방향 전파와 역방향 전파를 사용하는 함수의 자동 미분에 대해 간단하게 소개하려고 한다. 먼저 체인룰에 의해 다음과 같이 함수의 미분을 고려하자.

$$
\begin{aligned}
&y = f(g(h(x))) = f(g(h(w_0))) = f(g(w_1)) = x \\
&w_1 = h(w_0) \\
&w_2 = g(w_1) \\
&w_3 = f(w_2) = y
\end{aligned} \tag{1}
$$

$$
\frac{dy}{dx} = \frac{dy}{dw_2}\frac{dw_2}{dw_1}\frac{dw_1}{dx} = \frac{df(w_2)}{dw_2}\frac{dg(w_1)}{dw_1}\frac{dh(w_0)}{dx} \tag{2}
$$

2.1 순방향 전파

순방향 전파는 다음과 같은 식을 반복적으로 계산하여 설계 변수에 대한 목적함수의 미분값을 계산하는 것이다.

$$
\frac{dy}{dx} = \frac{dy}{dw_{n-1}}\left(\frac{dw_{n-1}}{dw_{n-2}}\left(\frac{dw_{n-2}}{dw_{n-3}}\frac{dw_{n-3}}{dw_x}\right)\right) \tag{3}
$$

예를 들어 $z = x_1 x_2 + \sin x_1$ 함수를 고려해 보자.

$$
\begin{aligned}
z &= w_1 w_2 + \sin w_1 \\
&= w_3 + w_4 = w_5
\end{aligned} \tag{4}
$$

순방향 전파에 따라 독립변수 x_1에 대해 순차적으로 미분값을 계산하면 다음과 같이 된다.

함수값 계산	미분값 계산
$w_1 = x_1$	$\dot{w}_1 = 1$
$w_2 = x_2$	$\dot{w}_2 = 0$
$w_3 = w_1 w_2$	$\dot{w}_3 = w_2\dot{w}_1 + w_1\dot{w}_2$
$w_4 = \sin w_1$	$\dot{w}_4 = \cos w_1 \dot{w}_2$
$w_5 = w_3 + w_4$	$\dot{w}_5 = \dot{w}_3 + \dot{w}_4$

n개의 벡터 변수에 의한 스칼라 함수에 대해 순방향 전파에 따른 계산 비용은 다음과 같은 것으로 제시되어 있다.

$$
1 \le \frac{Cost(f, \nabla f)}{Cost(f)} \le \phi(n) \tag{4}
$$

2.2 역방향 전파

역방향 전파는 민감도 해석의 보조변수법(Adjoint Variable Method)와 유사한 원리로 함수값의 계산을 완료한 다음 미분값을 계산하는 것으로 계산 결과를 저장하는 메모리를 필요로 하지만 설계 변수가 많을 경우에 더 효율적인 방법으로 알려져 있다. 역방향 전파의 경우 다음과 같이 최종 단계에서부터 반복적으로 미분값을 계산한다.

$$
\frac{dy}{dx} = \left(\left(\frac{\partial y}{\partial w_3}\frac{\partial w_3}{\partial w_2}\right)\frac{\partial w_2}{\partial w_1}\right)\frac{\partial w_1}{\partial w} \tag{5}
$$

동일하게 $z = x_1 x_2 + \sin x_1$ 함수를 고려해 보자. 역방향 전파에 따라 중간변수 w에 대한 미분값 $\overline{w} = \partial y / \partial w$를 계산하면 다음과 같이 된다.

$$
\begin{aligned}
z &= w_1 w_2 + \sin w_1 \\
&= w_3 + w_4 = w_5
\end{aligned} \tag{6}
$$

미분값 계산
$\overline{w_5} = 1$
$\overline{w_4} = \overline{w_5}\frac{\partial w_5}{\partial w_4} = \overline{w_5}$
$\overline{w_3} = \overline{w_5}\frac{\partial w_5}{\partial w_3} = \overline{w_5}$
$\overline{w_2} = \overline{w_3}\frac{\partial w_3}{\partial w_2} = \overline{w_3} w_1$
$\overline{w_1}^a = \frac{\partial y}{\partial w_4}\frac{\partial w_4}{\partial w_1} = \overline{w_4}\cos(w_1)$ $\overline{w_1}^b = \frac{\partial y}{\partial w_3}\frac{\partial w_3}{\partial w_1} = \overline{w_3} w_2$
$\overline{w_1} = \overline{w_1}^a + \overline{w_1}^b$ $\overline{w_2} = x_1$

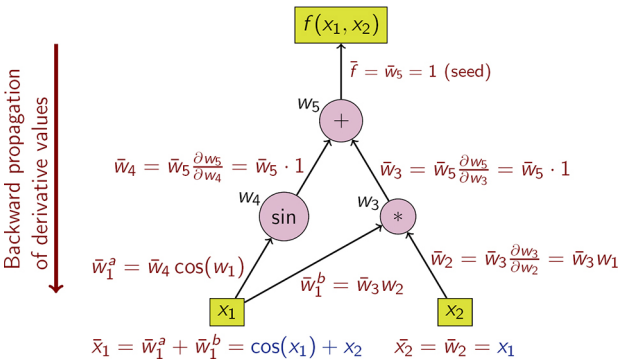

그림 1 역방향 전파를 이용한 자동 미분 (Wikipedia[4])

마찬가지로 n개의 벡터 변수에 의한 스칼라 함수에 대해 역방향 전파에 따른 계산 비용은 다음과 같다. 여기서 m은 종속 변수의 수를 의미한다.

$$1 \leq \frac{Cost(f, \nabla f)}{Cost(f)} \leq 1+3m$$

3. 자동 미분의 공학문제 적용 및 라이브러리

공학문제에서 자동 미분은 민감도 해석 또는 최적화에 활용되게 된다. 90년대 초반 자동 미분을 이용한 ADIFOR와 같은 Fortran 라이브러리가 개발된 이후 Barthlemy 등[5]은 트러스 최적화 문제에 자동 미분 알고리즘을 적용하여 수치적 성능을 비교하였다. 유사하게 Bishof 등[6] 또한 자동 미분을 이용하여 전산유체역학 문제에서 최적설계를 수행하였다.

흥미롭게도 위상최적설계 문제에서 자동 미분이 활용된 것은 비교적 최근이며 2016년 Nørgaard 등[7]이 비정상 유동의 위상최적설계 그리고 2021년 Chandrasekhar 등[8]이 전통적인 구조 위상최적설계에 자동 미분을 적용하였다. 자동 미분을 적용할 경우 계산시간은 해석적 민감도를 적용하는 경우와 차이가 크지 않은 것으로 보고되어 있다. 통상적인 구조 최적화에 사용되는 컴플라이언스를 목적 함수로 지정할 경우 해석적 민감도 식이 강성행렬을 포함한 식으로 구성되어 어렵지 않게 계산되므로 자동 미분을 적용하는 장점이 크지 않으나, 해석적 민감도 유도가 복잡한 응력제한조건을 가지는 위상최적설계의 경우 보다 쉽게 민감도를 계산할 수 있을 것이다.

최근 인공지능 및 딥러닝 분야의 비약적인 발전과 함께 자동 미분과 관련된 다수의 라이브러리가 개발되어 공개되고 있고, Tensorflow, PyTorch와 같은 범용 딥러닝 라이브러리에서 또한 기본적으로 신경망 학습을 위해서 자동 미분을 지원하고 있다.

표 1 Python 자동 미분 라이브러리

Library	Mode	URL
Autograd	F, R	https://github.com/HIPS/autograd
PyTorch	R	http://pytorch.org
Tangent	F, R	https://github.com/google/tangent
JAX	F, R	https://github.com/google/jax

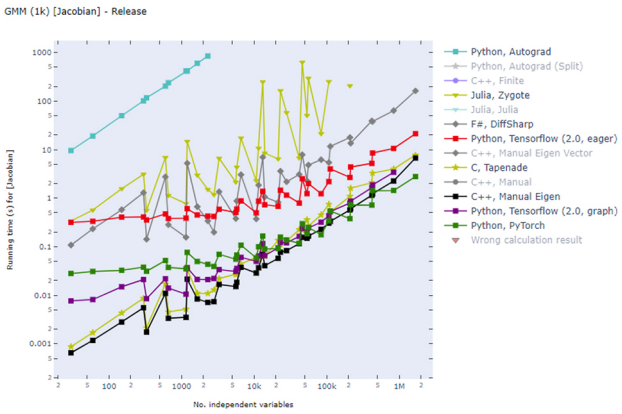

그림 2 자동 미분 라이브러리 별 계산시간 비교[9]

최근 인공지능 분야에서 가장 널리 사용되는 프로그래밍 언어는 Python이며 Python에서 지원하는 자동 미분 라이브러리를 표 1에 나타내었다.

자동 미분 라이브러리 별로 독립변수 수에 따른 실행시간을 비교한 자료를 그림 2에 나타내었다. 전반적으로 C, C++을 이용한 라이브러리의 성능이 우수한 것으로 나타났으나 PyTorch의 경우 독립변수 개수가 증가할 때 거의 유사한 성능을 보임을 확인할 수 있다.

라이브러리 별로 자동 미분의 성능을 비교할 때 한가지 유의할 점은 자동 미분에서 내부 선형방정식 계산 함수를 지원하는가이다. 인공지능 학습과는 달리 대부분의 공학문제는 선형방정식을 풀어야 하는 문제이며 라이브러리에서 지원하는 희소 행렬(sparse matrix) 계산 알고리즘이 어떠한 것인가에 따라 성능이 달라질 것이다.

4. 맺는 말

공학 문제의 최적화에서 다수의 알고리즘은 설계 변수에 대한 목적 함수의 경사도를 계산하는 것이 필요하며 빠른 경사도 계산을 위해 해석적 설계민감도를 유도하는 것 또한 많은 연구가 이루어져 왔다. 최근 머신 러닝 분야의 급속한 발전과 함께 자동 미분 알고리즘이 범용 프로그래밍 언어에 적용되었으며, 여러 분야의 최적화 문제에서 해석적 설계민감도 계산을 대체하는 방법으로 자동 미분 알고리즘이 적용 가능하다.

본 원고에서는 자동 미분을 공학 문제에 적용한 몇 가지 예와 최적설계에 적용 가능성을 확인하였다. 다수의 오픈소스 범용 프로그래밍 언어에서 자동 미분 라이브러리를

지원하는 것을 확인할 수 있지만 실제 대상이 되는 공학문제의 최적설계에 적용하기 위해서는 라이브러리 별 성능, 희소행렬 계산 지원 등을 고려할 필요가 있다. 또한 기존의 수치해석 교과목에서 유한 차분에 기반한 주제만을 다루고 있는데 향후 자동 미분 알고리즘을 강의 내용에 추가하는 것도 가능할 것이다.

참고문헌

1. R.E. Wengert (1964) A simple automatic derivative evaluation program. Comm. ACM. 7 (8), 463~464.
2. Seppo Linnainmaa (1976) Taylor Expansion of the Accumulated Rounding Error. BIT Numerical Mathematics. 16 (2), 146~160.
3. 이재훈, 김수환, 안중기, 권장혁 (2004) 자동미분의 공력 최적 설계 적용, 한국전산유체공학회, 춘계학술대회 논문집
4. Automatic differentiation from Wikipedia
5. J. -F. M. Barthelemy and L. E. Hall (1995) Automatic differentiation as a tool in engineering design. Structural optimization. 9, 76~82.
6. C. Bischof, G. Corliss, L. Green,A. Griewank, K. Haigler, P. Newman (1992) Automatic differentiation of advanced CFD codes for multidisciplinary design. Computing Systems in Engineering. 3 (6), 625~637.
7. Nørgaard, S. A., Sigmund, O., & Lazarov, B. S. (2016) Topology optimization of unsteady flow problems using the lattice Boltzmann method. Journal of Computational Physics, 307, 291~307.
8. A. Chandrasekhar and K. Suresh (2021) TOuNN: Topology Optimization using Neural Networks. Structural and Multidisciplinary Optimization 63, 1135~1149.
9. https://github.com/microsoft/ADBench