"How to Differentiate a Computer Program"

Daniel Grose

November 21, 2017

What AD is ...

What AD is ...

- Computer programs execute sequences of elementary arithmetic operations.

What AD is ...

- Computer programs execute sequences of elementary arithmetic operations.+,-,sin,exp,log,I,.....

What AD is ...

- Computer programs execute sequences of elementary arithmetic operations.+,-,sin,exp,log,I,.....
- The list of elementary functions is not very long.

What AD is ...

- Computer programs execute sequences of elementary arithmetic operations.+,-,sin,exp,log,l,.....
- The list of elementary functions is not very long.
- Each of these elementary arithmetic operations can easily be differentiated (in terms of each other).

What AD is ...

- Computer programs execute sequences of elementary arithmetic operations.+,-,sin,exp,log,l,.....
- The list of elementary functions is not very long.
- Each of these elementary arithmetic operations can easily be differentiated (in terms of each other).
- The sequence can be written using function composition e.g.

What AD is ...

- Computer programs execute sequences of elementary arithmetic operations.+,-,sin,exp,log,I,.....
- The list of elementary functions is not very long.
- Each of these elementary arithmetic operations can easily be differentiated (in terms of each other).
- The sequence can be written using function composition e.g.

$$log(sin(x) + cos(x)) \equiv (log \circ (+cos) \circ sin \circ I)(x)$$

What AD is ...

- Computer programs execute sequences of elementary arithmetic operations.+,-,sin,exp,log,I,.....
- The list of elementary functions is not very long.
- Each of these elementary arithmetic operations can easily be differentiated (in terms of each other).
- The sequence can be written using function composition e.g.

$$log(sin(x) + cos(x)) \equiv (log \circ (+cos) \circ sin \circ I)(x)$$

- Sequences of function compositions can be differentiated using the chain rule.

What AD is ...

- Computer programs execute sequences of elementary arithmetic operations.+,-,sin,exp,log,I,.....
- The list of elementary functions is not very long.
- Each of these elementary arithmetic operations can easily be differentiated (in terms of each other).
- The sequence can be written using function composition e.g.

$$log(sin(x) + cos(x)) \equiv (log \circ (+cos) \circ sin \circ I)(x)$$

- Sequences of function compositions can be differentiated using the chain rule.
- This has been automated - it is called **A**lgorithmic (or **A**utomatic) **D**ifferentiation, or **AD** for short.

What AD is not ...

What AD is not ...

- Numerical differentiation e.g finite differences etc.

What AD is not ...

- Numerical differentiation e.g finite differences etc.
- Symbolic differentiation (other than a small number of elementary functions).

What AD is not ...

- Numerical differentiation e.g finite differences etc.
- Symbolic differentiation (other than a small number of elementary functions).
- An approximation - it is as accurate as the compiler can allow.

What AD is not ...

- Numerical differentiation e.g finite differences etc.
- Symbolic differentiation (other than a small number of elementary functions).
- An approximation - it is as accurate as the compiler can allow.

What AD is good at ...

What AD is good at ...

- **AD** is at its best for functions $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.

What AD is good at ...

- **AD** is at its best for functions $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, particularly when n and/or m is large.
- Rapid development - no need to code the $m \sum_{i=1}^{k} \left(\!\binom{n}{i}\!\right) = m \sum_{i=1}^{k} \binom{n+i-1}{i}$ functions needed to compute the partial derivatives to k_{th} order.

What AD is good at ...

- **AD** is at its best for functions $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, particularly when n and/or m is large.
- Rapid development - no need to code the $m \sum_{i=1}^{k} \left(\!\binom{n}{i}\!\right) = m \sum_{i=1}^{k} \binom{n+i-1}{i}$ functions needed to compute the partial derivatives to k_{th} order. Example - $n = 4$, $m = 3$ and $k = 2$ requires 42 additional functions !!

What AD is good at ...

- **AD** is at its best for functions $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Rapid development - no need to code the $m\sum_{i=1}^{k}\left(\!\binom{n}{i}\!\right) = m\sum_{i=1}^{k}\binom{n+i-1}{i}$ functions needed to compute the partial derivatives to k_{th} order. Example - $n = 4$, $m = 3$ and $k = 2$ requires 42 additional functions !!
- Efficiency - for $k = 2$, **AD** takes the equivalent of about 4 evaluations of f.

What AD is good at ...

- **AD** is at its best for functions $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, particularly when n and/or m is large.
- Rapid development - no need to code the $m\sum_{i=1}^{k}\left(\!\binom{n}{i}\!\right) = m\sum_{i=1}^{k}\binom{n+i-1}{i}$ functions needed to compute the partial derivatives to k_{th} order. Example - $n = 4$, $m = 3$ and $k = 2$ requires 42 additional functions !!
- Efficiency - for $k = 2$, **AD** takes the equivalent of about 4 evaluations of f. With $n = 4$, $m = 3$ and $k = 2$, finite differences would require <u>243</u> evaluations of f.

What AD is good at ...

- **AD** is at its best for functions $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Rapid development - no need to code the $m \sum_{i=1}^{k} \left(\!\binom{n}{i}\!\right) = m \sum_{i=1}^{k} \binom{n+i-1}{i}$ functions needed to compute the partial derivatives to k_{th} order. Example - $n = 4$, $m = 3$ and $k = 2$ requires 42 additional functions !!
- Efficiency - for $k = 2$, **AD** takes the equivalent of about 4 evaluations of f. With $n = 4$, $m = 3$ and $k = 2$, finite differences would require <u>243</u> evaluations of f.
- Accuracy - machine precision.

What AD is good at ...

- **AD** is at its best for functions $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Rapid development - no need to code the $m \sum_{i=1}^{k} \left(\!\binom{n}{i}\!\right) = m \sum_{i=1}^{k} \binom{n+i-1}{i}$ functions needed to compute the partial derivatives to k_{th} order. Example - $n = 4$, $m = 3$ and $k = 2$ requires 42 additional functions !!
- Efficiency - for $k = 2$, **AD** takes the equivalent of about 4 evaluations of f. With $n = 4$, $m = 3$ and $k = 2$, finite differences would require <u>243</u> evaluations of f.
- Accuracy - machine precision.

What AD is useful for ...

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation.

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, ...) = 0$?

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, ...) = 0$? When does $f'(\mathbf{x}, \mathbf{y}, ...) = 0$?

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, ...) = 0$? When does $f'(\mathbf{x}, \mathbf{y}, ...) = 0$? Newton–Raphson.

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, ...) = 0$? When does $f'(\mathbf{x}, \mathbf{y}, ...) = 0$? Newton–Raphson.
- Sensitivity / Robustness tests.

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, \ldots..) = 0$? When does $f'(\mathbf{x}, \mathbf{y}, \ldots..) = 0$? Newton–Raphson.
- Sensitivity / Robustness tests. If $f'(\mathbf{x}, \mathbf{y}) = 0$ what is $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$?

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, ...) = 0$? When does $f'(\mathbf{x}, \mathbf{y}, ...) = 0$? Newton–Raphson.
- Sensitivity / Robustness tests. If $f'(\mathbf{x}, \mathbf{y}) = 0$ what is $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$? Implicit Function Theorem.

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, ...) = 0$? When does $f'(\mathbf{x}, \mathbf{y}, ...) = 0$? Newton–Raphson.
- Sensitivity / Robustness tests. If $f'(\mathbf{x}, \mathbf{y}) = 0$ what is $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$? Implicit Function Theorem.
- Quants love the last one !!

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \to \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, ...) = 0$? When does $f'(\mathbf{x}, \mathbf{y}, ...) = 0$? Newton–Raphson.
- Sensitivity / Robustness tests. If $f'(\mathbf{x}, \mathbf{y}) = 0$ what is $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$? Implicit Function Theorem.
- Quants love the last one !! Perhaps the biggest application area.

What AD is useful for ...

- Jacobians, Hessians and higher order derivatives of $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, particularly when n and/or m is large.
- Optimisation. When does $f(\mathbf{x}, \mathbf{y}, ...) = 0$? When does $f'(\mathbf{x}, \mathbf{y}, ...) = 0$? Newton–Raphson.
- Sensitivity / Robustness tests. If $f'(\mathbf{x}, \mathbf{y}) = 0$ what is $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$? Implicit Function Theorem.
- Quants love the last one !! Perhaps the biggest application area. Not much in public domain !!

Time permitting ... An example using R

Time permitting ... An example using R

- Start R.

Time permitting ... An example using R

- Start R. Load library.

Time permitting ... An example using R

- Start R. Load library. Define function.

Time permitting ... An example using R

- Start R. Load library. Define function. Use function.

Time permitting ... An example using R

- Start R. Load library. Define function. Use function.

```
> library(adlaComp)
> f<-adlaComp('
adlaMat f(const adlaMat& X)
{
   Eigen::EigenSolver<adlaMat> es(X);
   return es.pseudoEigenvectors();
}
')
> x<-matrix(c(1,2,3,4),2,2)
> f(x)
           [,1]       [,2]
[1,] -0.9093767 -0.4159736
[2,]  0.4159736 -0.9093767
```

Time permitting ... An example using R

- Calculate Jacobian.

Time permitting ... An example using R

- Calculate Jacobian.

```
> J(f)(x)
            [,1]        [,2]        [,3]        [,4]
[1,]  0.02739166 -0.01252969  0.05988202 -0.02739166
[2,] -0.05988202  0.02739166 -0.13091051  0.05988202
[3,]  0.05988202 -0.02739166  0.13091051 -0.05988202
[4,]  0.02739166 -0.01252969  0.05988202 -0.02739166
```

Time permitting ... An example using R

- Calculate (stacked) Hessian.

Time permitting ... An example using R

- Calculate (stacked) Hessian.

```
> H(f)(x)
             [,1]         [,2]          [,3]         [,4]          [,5]
[1,]  0.009551480 -0.006550244  0.0104567584 -0.009551480 -0.0104567584
[2,] -0.006550244  0.003993971 -0.0095514802  0.006550244  0.0095514802
[3,]  0.010456758 -0.009551480  0.0000713503 -0.010456758 -0.0000713503
[4,] -0.009551480  0.006550244 -0.0104567584  0.009551480  0.0104567584
             [,6]          [,7]          [,8]          [,9]        [,10]
[1,]  0.009551480 -0.0000713503  0.0104567584  0.0104567584 -0.009551480
[2,] -0.006550244  0.0104567584 -0.0095514802 -0.0095514802  0.006550244
[3,]  0.010456758  0.0496630913  0.0000713503  0.0000713503 -0.010456758
[4,] -0.009551480  0.0000713503 -0.0104567584 -0.0104567584  0.009551480
             [,11]         [,12]        [,13]        [,14]         [,15]
[1,]  0.0000713503 -0.0104567584  0.009551480 -0.006550244  0.0104567584
[2,] -0.0104567584  0.0095514802 -0.006550244  0.003993971 -0.0095514802
[3,] -0.0496630913 -0.0000713503  0.010456758 -0.009551480  0.0000713503
[4,] -0.0000713503  0.0104567584 -0.009551480  0.006550244 -0.0104567584
            [,16]
[1,] -0.009551480
[2,]  0.006550244
[3,] -0.010456758
[4,]  0.009551480
```