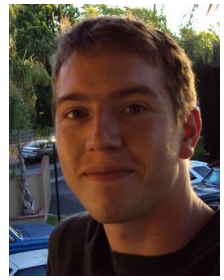
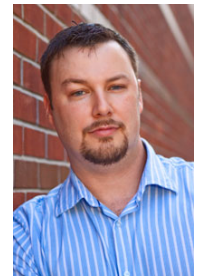
Autograd: Effortless gradients in Pure Python

Dougal Maclaurin, David Duvenaud, Ryan Adams

HARVARD
School of Engineering
and Applied Sciences

Motivation

- Gradients are hard to derive and code correctly
- Wish we could write whatever complicated Python & Numpy code, and get gradients automatically
- Also: Higher derivatives for Hessian-vector products

Autograd: Automatic Differentiation

- `github.com/HIPS/autograd`
- Simple ($\sim$ 300 lines of code)
- Functional interface
- Works with (almost) arbitrary Python/numpy code
- Can take gradients of gradients (of gradients...)

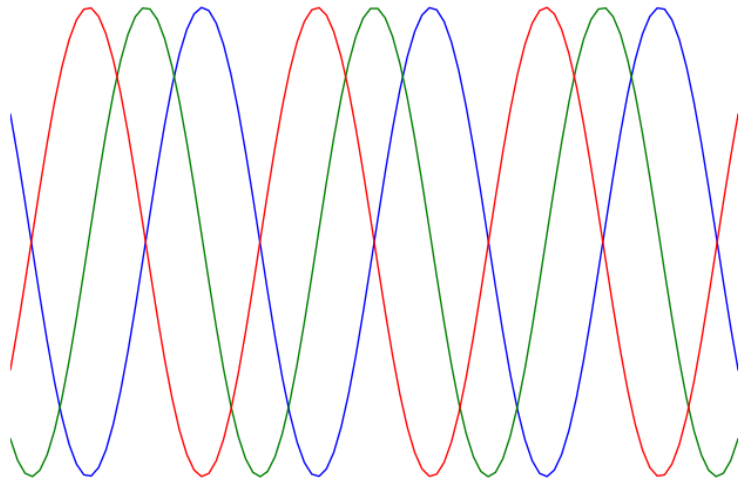
Autograd Examples

```
import autograd.numpy as np
import matplotlib.pyplot as plt
from autograd import grad

def fun(x):
    return np.sin(x)

d_fun = grad(fun)     # First derivative
dd_fun = grad(d_fun)  # Second derivative

x = np.linspace(-10, 10, 100)
plt.plot(x, map(fun, x),
         x, map(d_fun, x),
         x, map(dd_fun, x))
```



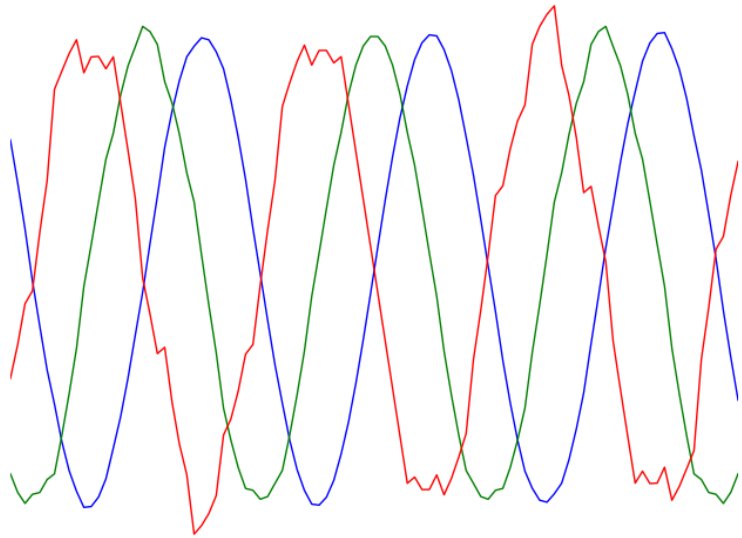
Autograd Examples

```
import matplotlib.pyplot as plt
import autograd.numpy as np
from autograd import grad

# Taylor approximation to sin function
def fun(x):
    curr = x
    ans = curr
    for i in xrange(1000):
        curr = - curr * x**2 / ((2*i+3)*(2*i+2))
        ans = ans + curr
        if np.abs(curr) < 0.2: break
    return ans

d_fun = grad(fun)
dd_fun = grad(d_fun)

x = np.linspace(-10, 10, 100)
plt.plot(x, map(fun, x),
         x, map(d_fun, x),
         x, map(dd_fun, x))
```



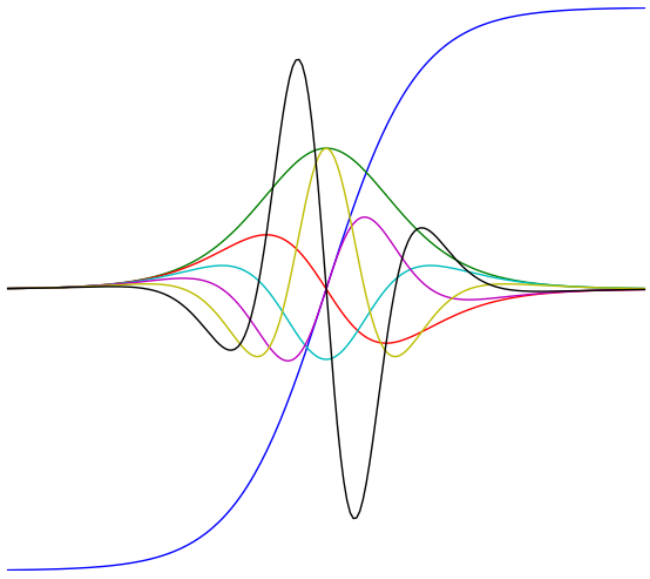
Autograd Examples

```
import matplotlib.pyplot as plt
import autograd.numpy as np
from autograd import grad

def tanh(x):
    return (1 - np.exp(-x)) / (1 + np.exp(-x))

d_fun = grad(tanh)             # 1st derivative
dd_fun = grad(d_fun)           # 2nd derivative
ddd_fun = grad(dd_fun)         # 3rd derivative
dddd_fun = grad(ddd_fun)       # 4th derivative
ddddd_fun = grad(dddd_fun)     # 5th derivative
dddddd_fun = grad(ddddd_fun)   # 6th derivative

x = np.linspace(-7, 7, 200)
plt.plot(x, map(tanh, x),
         x, map(d_fun, x),
         x, map(dd_fun, x),
         x, map(ddd_fun, x),
         x, map(dddd_fun, x),
         x, map(ddddd_fun, x),
         x, map(dddddd_fun, x))
```



Most Numpy functions implemented

Complex & Fourier	Array	Misc	Linear Algebra	Stats
imag	atleast_1d	logsumexp	inv	std
conjugate	atleast_2d	where	norm	mean
angle	atleast_3d	einsum	det	var
real_if_close	full	sort	eigh	prod
real	repeat	partition	solve	sum
fabs	split	clip	trace	cumsum
fft	concatenate	outer	diag	
fftshift	roll	dot	tril	
fft2	transpose	tensordot	triu	
ifftn	reshape	rot90		
ifftshift	squeeze			
ifft2	ravel			
ifft	expand_dims			
fftn	flipud			

More Autograd Examples

- Fully-connected neural net
- Convolutional neural net
- Recurrent neural net
- LSTM
- Population genetics simulations

```
pip install autograd
```