PA1 Writeup

2.3

Problem

The starter implementations of BA_Add.grad_fn (and similarly Sub, Mul, Div) assume that inputs x and y always have the same shape or are scalars. This fails when broadcasting occurs (e.g., x.shape = (3,1) and y.shape = (1,4)), because the upstream gradient will have a larger shape than the original inputs. Directly adding self.grad leads to shape mismatch errors.

Fix

We introduced an unbroadcast helper that reduces the gradient back to the original shape by summing along broadcasted dimensions. Each grad_fn now uses this helper to correctly align the gradient with the shape of the original inputs.

2.4

Testing Method

We compared numerical differentiation (central difference with eps=1e-5) and automatic differentiation (backpropagation) on two functions, g1 and g2. Tests were performed at x = 0.5.

Results

=== Testing g1 === g1: numerical = 9.000000000014552 autodiff = 9.0

=== Testing g2 === g2: numerical = 377.79462820850534 autodiff = 377.79462821241566

Analysis

- For g1, both methods give ~9, with a negligible difference (~1e-11).
- For g2, both methods give ~377.79, with a negligible difference (~1e-9).
- These small discrepancies are due to floating-point precision.

Conclusion

Automatic differentiation matches numerical differentiation within floating-point error, confirming the correctness of the implementation.

2.6

Testing Method

To perform the test, we construct an array [1.0, 2.0, 3.0, 4.0, 5.0] and compute its gradient under both the numerical gradient function (with epsilon equals to 1e-5) and automatic differentiation.

Results

=== numerical_diff === numerical: [0. -4. 8. 48. 127.99999999] === backprop_diff === backprop: [0. -4. 8. 48. 128.]

Analysis

From the results, we observe that, except for the last value in the array, all other values in numerical differentiation and backpropagation are identical. The difference in the last value between these two implementations is very small (1e-08), which confirms the correctness of our numerical implementation.

2.7

From the results, we find that the time elapsed for automatic differentiation is around 0.09 ms, whereas for numerical differentiation it is around 11.16 ms, which is more than 100 times longer. Therefore, the backprop differentiation is much faster than numerical differentiation.