Exercise Set 4: Solutions

# Main exercise

**Here is my implementation:**

float FDerivative(float x)

{

// dg/dx = dg/dh \* dh/di \* di/dx

float di\_dx = IDerivative(x);

float i = I(x);

float dh\_di = HDerivative(i);

float h = H(i);

float dg\_dh = GDerivative(h);

return dg\_dh \* dh\_di \* di\_dx;

}

**For F being a sum of two nested functions, I changed F to be this:**

float F(float x)

{

return G(H(I(x))) + G(I(x));

}

**And I changed FDerivative to this:**

float FDerivative(float x)

{

// The left side

float left = 0.0f;

{

float di\_dx = IDerivative(x);

float i = I(x);

float dh\_di = HDerivative(i);

float h = H(i);

float dg\_dh = GDerivative(h);

left = dg\_dh \* dh\_di \* di\_dx;

}

// The right side

float right = 0.0f;

{

float di\_dx = IDerivative(x);

float i = I(x);

float dg\_di = GDerivative(i);

left = dg\_di \* di\_dx;

}

return left + right;

}

# Answers: More things to try

When playing with the functions, did you notice the values ever shooting off to infinity despite the function definitely having a global minimum? This is “exploding gradients” — a known problem that comes up in deep learning along with “vanishing gradients,” which makes the gradient go to zero (the graph goes flat). Both problems come up in deep learning situations *specifically* because deep learning deals with many-layered neural networks.

If you’d like an introduction on dealing with exploding and vanishing gradient problems, start here: <https://machinelearningmastery.com/exploding-gradients-in-neural-networks/>