As a simple example, take a network with 1 input layer, 1 hidden layer, and 1 output layer . The hidden and output layers use sigmoid activation, $\frac{1}{1+e^{-x}}$, referred to here as $\sigma(x)$

Let a^L refer to the output of layer L after activation

Let $s^L = w^L \cdot a^{L-1} + b^L$, such that $a^L = \sigma(s^L)$

Let $E = (a^L - y)^2$ for some target y

Then,

$$\begin{aligned}\frac{\partial E}{\partial w^L} &= \frac{\partial s^L}{\partial w^L}\frac{\partial a^L}{\partial s^L}\frac{\partial E}{\partial a^L} \\ &= a^{L-1} \cdot \sigma(s^L)(1 - \sigma(s^L)) \cdot 2(a^L - y) \\ &= a^{L-1} \cdot a^L(1 - a^L) \cdot 2(a^L - y)\end{aligned}$$

And the result of this equation is the value by which the weight connecting the hidden layer and output layer needs to change in order to most efficiently reduce the the cost.

I think this is correct, however my confusion comes from when we have mutltiple neurons in each layer. If I instead have a network with 3 layers, an input layer with 3 neurons, a hidden layer with 3 neurons, and an output layer with 2 neurons, with the hidden and output using the same activation as before, σ, how does the math involved in back propagation change?

Now,

$E = \sum_{i=0}^{n}(a_i^L - y_i)^2$ for n neurons in the output layer (I think)

But how is the partial derivative of that expression calculated with respect to the weight?