* Each neuron has a bias.
* Biases are learnable.
* Bias determines if a neuron is activated
* Bias increases the flexibility of the model

While the weights and biases are both types of learnable parameters in the network, they influence the network in different ways. For example, changing the values for the weights can influence where we fall on the graph of the activation function, say relu, for a particular layer, but changing the value for the biases will change the position of the graph of relu all-together. The response for (3) elaborates more on this.

Yes, the weights and biases are being updated at the same time using SGD and backpropagation. It’s not necessarily happening in the same calculation, but it is happening in the same step. Just as we saw in the backprop videos earlier in this playlist, SGD calculates the gradient of the loss with respect to the weights (via backprop) and then updates the weights with the result. For bias, SGD similarly does this same process of calculating the gradient, but with respect to the biases rather than the weights.

That’s one way of thinking about it-- as some sort of compensation for the inflexibility of the activation function. I’m not sure of a “more flexible” non-linear activation function that exists and has been adopted for use in neural networks. Relu is pretty much the go-to standard for now. If you think about the graph of relu though, you could actually think of it as maybe being pretty flexible. I mean, it's a linear function for all numbers greater than or equal to zero, so it spans the entire positive number line. But when we think about adding bias, we can think of the entire graph of relu shifting to the left or to the right. We’re handing over flexiblity to the training algorithm to decide what it should mean for a neuron to be meaningfully activated, rather than just saying, “if you’re greater than zero, you’re active” and also having it decide different levels of activations for different neurons all using the same underlying activation function.