Discussion L2b - abbreviated

1) What implications on tuning weights can have chaining of multiplications in the calculation of the error derivative wrt. weights close to the input (nested dependencies results in chaining of derivatives in backprop)?

2) What does the node saturation mean and imply? How can it manifest itself? Why do we initialize weights exploiting zero-mean distributions?

3) How would you choose the network size? What are differences or implications of large vs small networks? What do you think are the consequences of deep architectures (more hidden layers and nodes) for the error surface?

4) Momentum term is often thought of as an inertia force – can you try to explain why? In what situations is it useful (compare “flat” areas on the error surface with “sinusoidal” downhills) and how?

5) Learning rate is one of the most important hyperparameter besides the size of the network. How could you choose a suitable learning rate or automatically adapt it in backprop?

Discussion L2b - supplementary

1) How do you know if the compression with an autoencoder is lossless (or lossy)? What would happen if the hidden layer in the autoencoder was of larger dimensionality than the input-output? What sort of data representation could be expected as opposed to the representation/coding we get with hour-glass architecture?

2) You are supposed to “model” a control system for a nuclear plant (there are some control variables that affect a multi-dimensional energy output) in a data-driven manner without knowing much about the underlying mechanisms other than it is a highly nonlinear system. How would you approach the problem with an MLP, i.e. choose parameters, train and test it etc.? What would you pay special attention to or be concerned about regarding the available data for training but also the validation (can we rely on it)? Could you use your MLP model to control a similar plant?

3) Could we use an MLP for time series prediction? Let’s say we want to predict a stock market (some financial time series) - what would be inputs and outputs? How would you train it? What activation function would you choose for hidden layers and/or outputs?