## 2.1

### a

A nonlinear transfer function gives the neural network a universal property: Given enough layers and neurons, the network can model any function within a certain accuracy. In a network with a linear transfer function we can only compute a linear function. A network with a linear transfer function of $n$ layers will always be equivalent to a network with only one layer: linear functions can always be concatinated.

Whenever the function that we are trying to model isn't a linear function, it is useful to use a nonlinear transfer function. Examples include image classification or speech recognition.

### b

Consider a simple neural network with two input neurons that can both either be 0 or 1 and one output layer. We want to construct an AND gate with our network, so without bias our quest would be to find $w = (w_1, w_2)$ such that:

$$0 \leq 0$$
$$w_1 \leq 0$$
$$w_2 \leq 0$$
$$w_1 + w_2 > 0$$

which is impossible. We can easily however create the network with a bias, if we have the weights $w = (1, 1)$ and the bias $\theta = \frac{3}{2}$. Then $sgn(w^T x - \theta)$ would give us AND.

### c

Point and edge filters are for example a connectionist neuron which gets values of a scalar field as input, that represent the color of each pixel or the color gradient or even a higher derivative and has weights in the following form: In the simple case of two colors (0 and 1) this point filter would return



point filter          edge filter (Sobel filter)

zero for no point, 1 or -1 for a point in the outer region and 8 or -8 if the point is in the middle. This goes analogously for the other filter.

**d**

The first is deterministic and the second has a noise parameter and can
return different states for set parameters and a given input.

## 2.2

## 2.3

**a**

A MLP could decide between a horizontal and a vertical edge, whereas a
perceptionist neuron would either be able to differ between vertical edge or
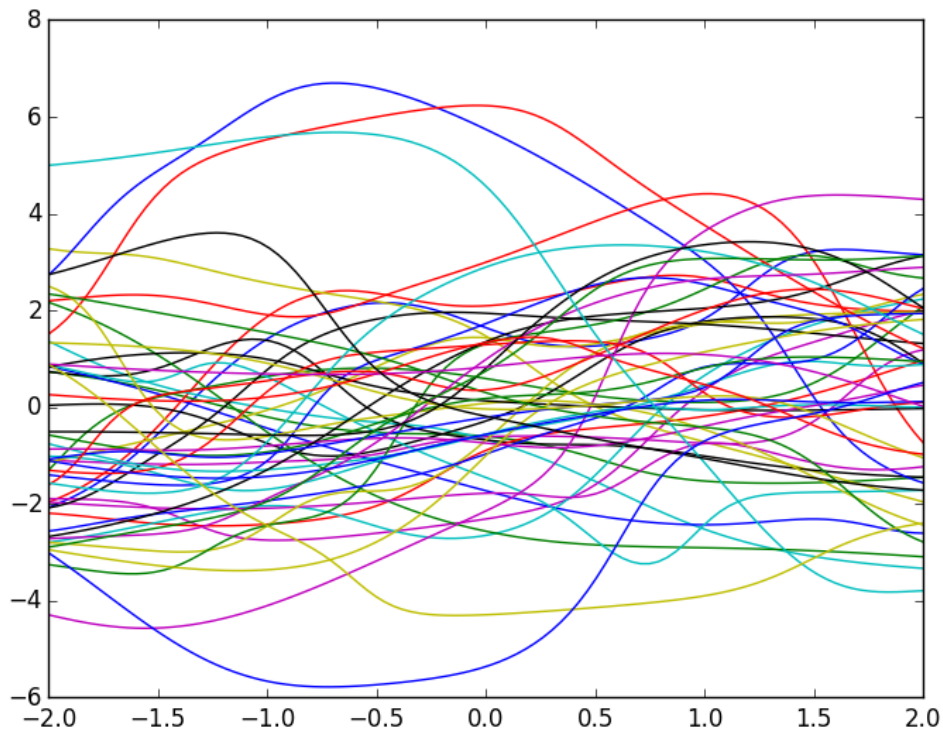no vertical edge OR horizontal edge or no horizontal edge.

**b**



Figure 1: functions computed with normally distributed $a_i$ with a standard
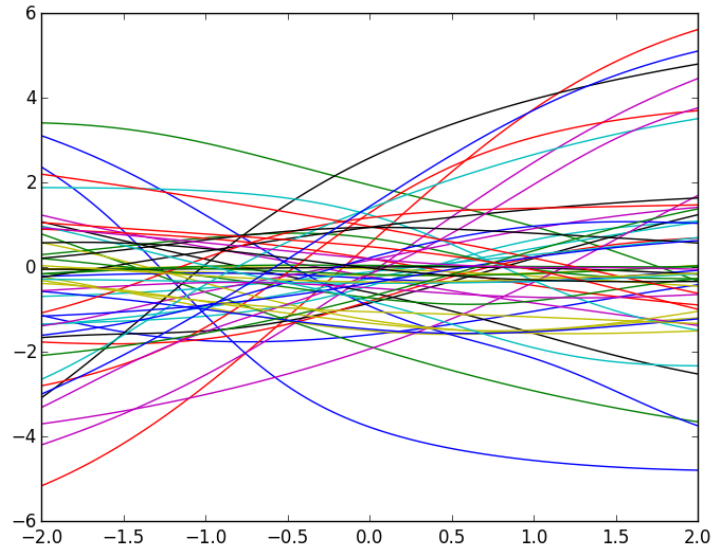deviation of 2

**c + bonus**



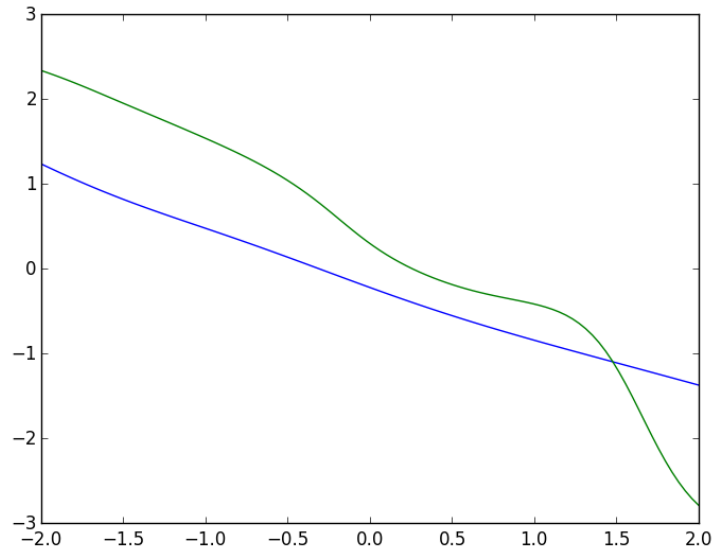Figure 2: functions computed with normally distributed $a_i$ with a standard deviation of 0.5



Figure 3: functions with least mean square error from f(x)=-x, computed with an std of 2 (green) and 0.5 (blue)