## 3.1

**First question**. For RBFs I divide them uniformely around the input space (means pi/n and std deviation (mean\_n-mean\_n-1)/2

To reduce the error, we need:

|  |  |  |
| --- | --- | --- |
|  | Sin(2x) | Sq(2x) |
| 0.1 | 38-40 RBF | 38-40 RBF |
| 0.01 | 48-50 RBF | 60-63 RBF |
| 0.001 | 60-63 RBF | NOP (only 63) RBF |

¿Why validation error increases after one point (around RBF = nsamples/3)? Maybe because we don’t very many data compared to the number of RBFs

**Second question**. It can be solved like a classification problem, where all positive mapped to 1 and all negative mapped to 0 (threshold = 0). It can be solved with a single layer perceptron. We can do it with **at least 8 nodes**, since with less nodes some values in which square(2x) = 1 are negative with the approximation (SOMETIMES, DEPENDING OF THE NUMBER OF RBFS. **However, it works well (gets 0 errors) only with multiples of 4 (I AM TAKING INTO ACCOUNT TRAINING + VALIDATION DATA).** This transformation could be useful, for example, when you need to classify data, but input samples are noisy.

## 3.2

Best: 8-12 nodes

**First Question and Fift Question:** we should use the validation error, since the training error has a very noisy aspect in sequential case. The noisy aspect appears because the error is estimated used only one sample. We have compared delta rule with LS in terms of absolute error for sigma = 1.2 and among 2 and 16 RBFs. With clean data LS is better but for noisy data delta rule is better for (from 8 nodes to 16, before it they are similar).

**Second Question:** in the case of LS does not make sense to speak about learning rate since it obtains the optimum w in one step.

**Third Question:** the key here is that for a given number of RBF we should try to cover the whole input space with those RBFs. The first approach was to adapt the width depending on the number of RBFs (mean(1)-mean(2)/2). If the width is so high all neurons are activated independently of the input, while if we have very little neurons they are only activated for a very limited range of inputs. So the best way is to find a trade off and look for an appropriate width. In our case, the best configuration was found for around 8-14 RBFs and sigma >= 1.1

**Fourth Question:** selecting random means gives less stability, since the error depends on the random position. However, as an average, the error is the same that when means are selected manually.

**Sixth Question:** MLP is worse and also takes more time. TEST WITH DIFFERENT NOISE IN THE DATA!! (sigma = 0.1 and sigma = sqrt(0.1))