Least Mean Squares and Experiments with Non-Linearity

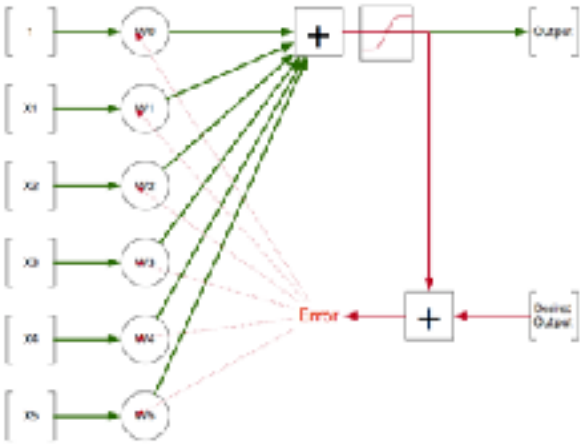
Apoorva Desai - Lab 1

Abstract

For Lab 1, a simple one layer neural net and the LMS algorithm were recreated without the use of any toolkits. With the LMS algorithm we predict the next value by producing the least mean square of the error from the previous values.

Approach

LMS is a linear regression algorithm. We begin by randomly initializing a weight vector with the same dimensions as the input vector. A dot product of these two vectors gives us the prediction of the output. Error is calculated as the difference of the actual output with the predicted value. This error is used to modify the weight vector. Input samples are created by windowing through the input audio sequence.



By introducing non-linearity we are basically emulating a single layer neural net without any hidden layers. A non-linearity activation function is applied to the predicted output value before updating the weights.

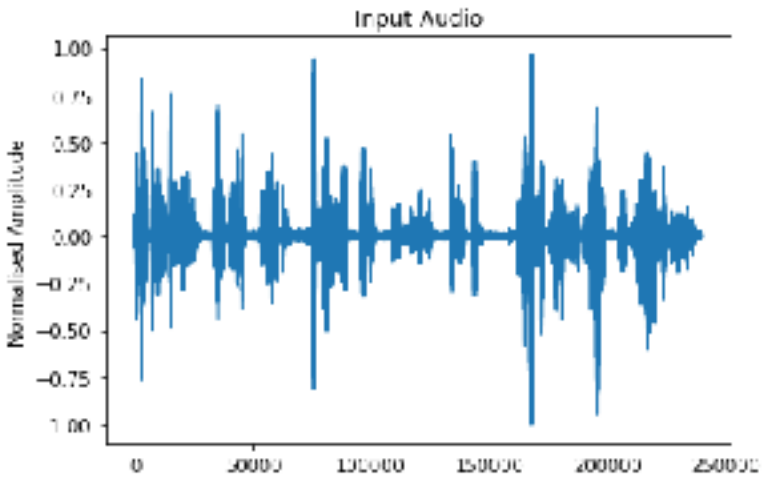
The image on the right depicts a single layer neural network. This could also be thought of as the LMS algorithm with non-linearity. There are no hidden layers.

Steps for audio manipulation:

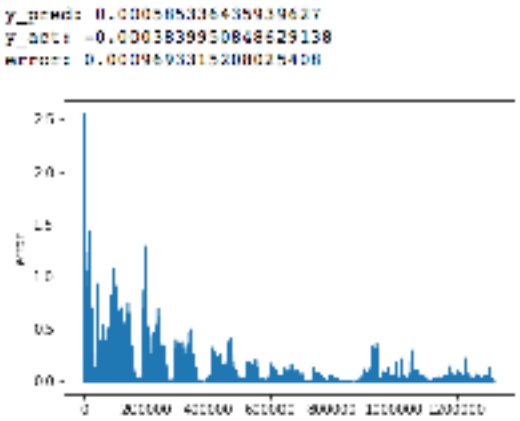
- The input speech and audio signal are both sampled at first 44.1kHz and then 8kHz.
- They are clipped to the same length and amplitude modified- 0.8 for the clean speech and 0.2 for the noise.
- They are then combined to create a noisy signal.

Results

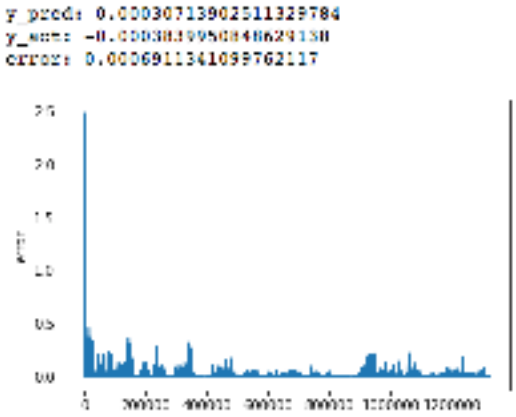
Input Audio Signal:



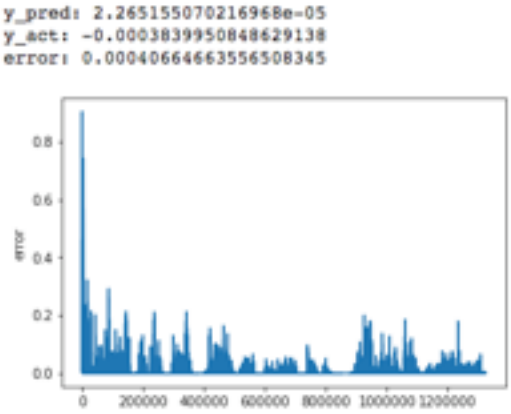
1. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 500.



2. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 100.

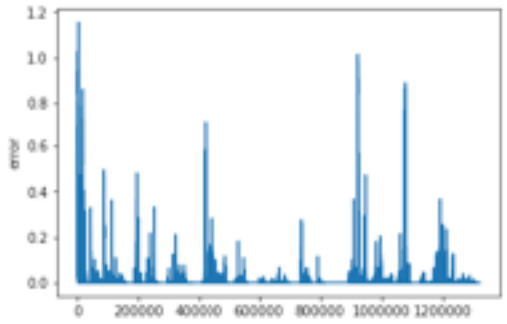


3. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 16.



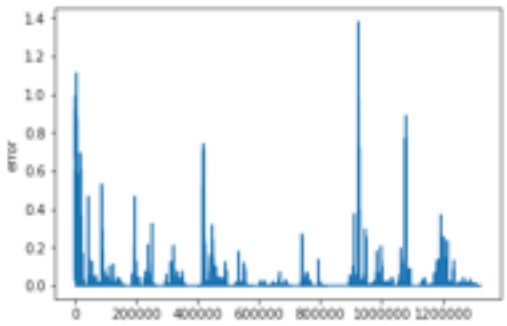
4. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 500 and Sigmoid Activation function.

```
y_pred: 0.005348949732511302
y_act: -0.0003839950848629138
error: 0.005732944817374215
```

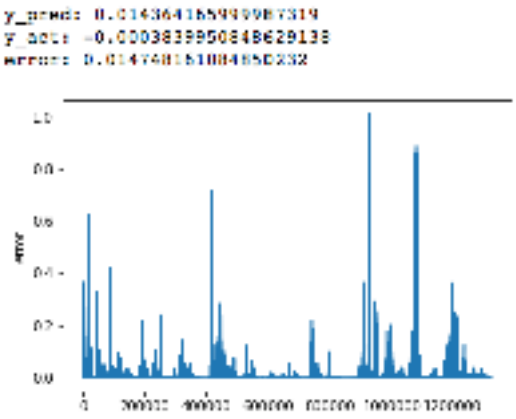


5. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 100 and Sigmoid Activation function.

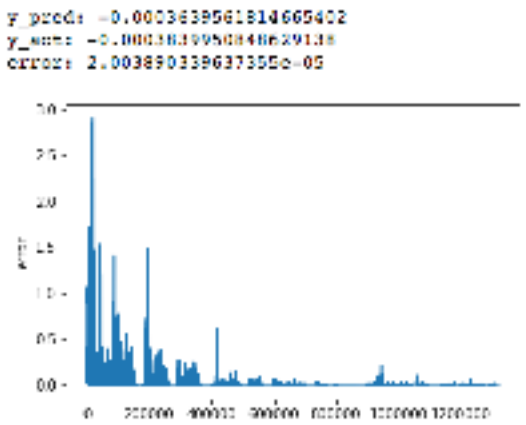
```
y_pred: 0.010323211095606761
y_act: -0.0003839950848629138
error: 0.010707206180469674
```



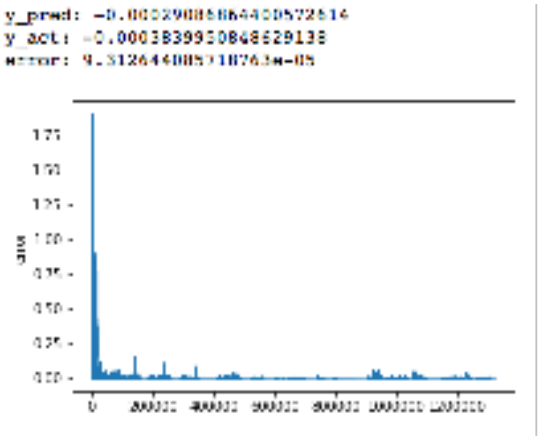
6. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 16 and Sigmoid Activation function.



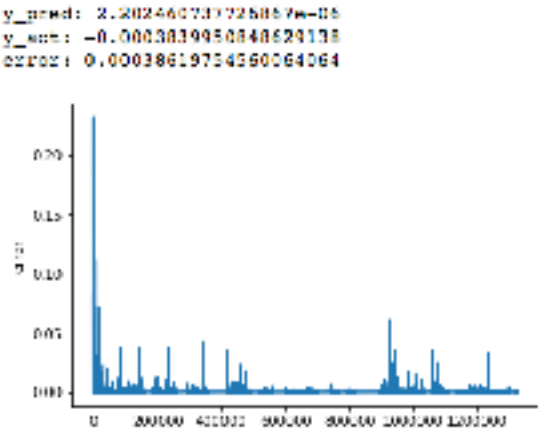
7. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 500 and tanh Activation function.



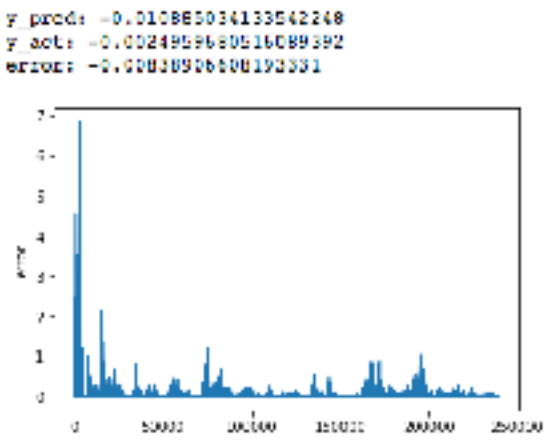
8. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 100 and tanh Activation function.



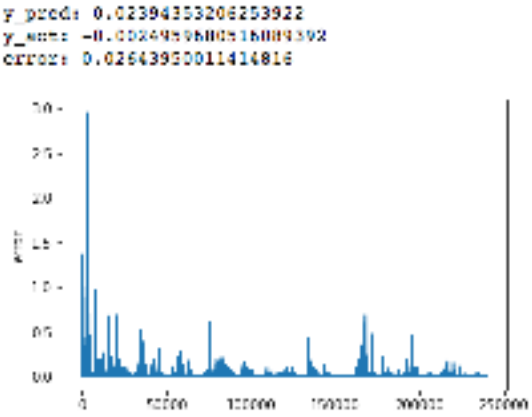
9. LMS algorithm on Clean Speech sampled at 44.1kHz window size of 16 and tanh Activation function.



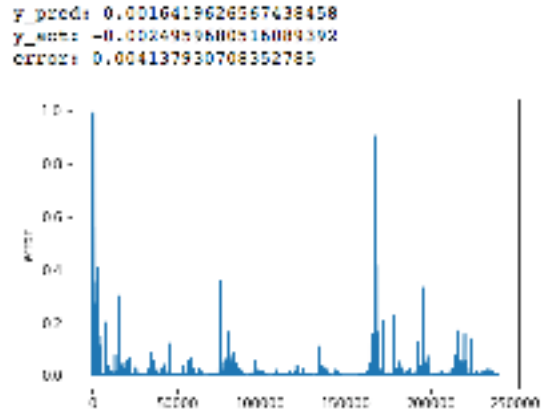
10. LMS algorithm on Clean Speech and Noisy Signal sampled at 8kHz window size of 500.



11. LMS algorithm on Clean Speech and Noisy Signal sampled at 8kHz window size of 100.

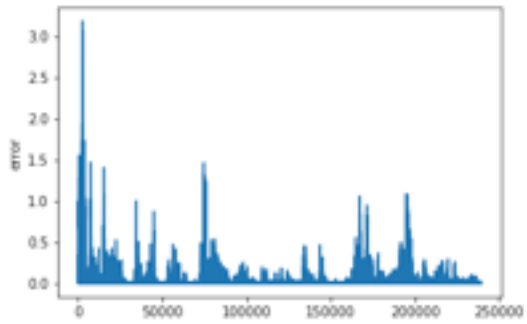


12. LMS algorithm on Clean Speech and Noisy Signal sampled at 8kHz window size of 16.



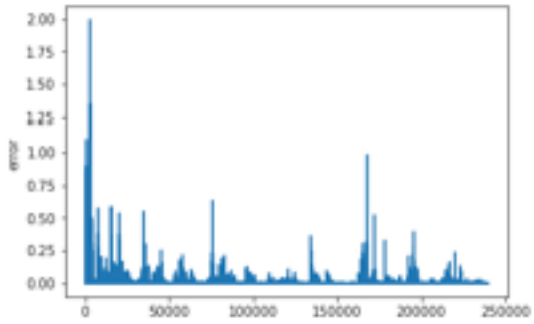
13. LMS algorithm on Clean Speech and Noisy Signal sampled at 8kHz window size of 500 and tanh Activation function.

```
y_pred: -0.006867156124855805
y_act: -0.0024959680516089392
error: -0.004371188073246865
```



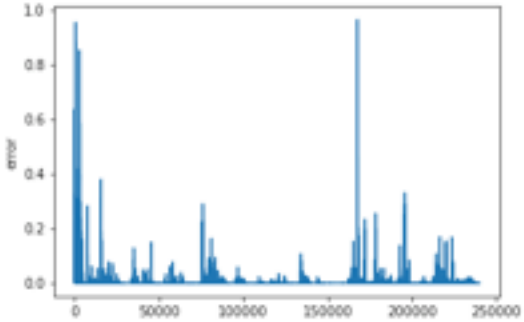
14. LMS algorithm on Clean Speech and Noisy Signal sampled at 8kHz window size of 100 and tanh Activation function.

```
y_pred: 0.022911072111084858
y_act: -0.0024959680516089392
error: 0.025407040162693798
```



15. LMS algorithm on Clean Speech and Noisy Signal sampled at 8kHz window size of 16 and tanh Activation function.

```
y_pred: -0.0015960974929355458
y_act: -0.0024959680516089392
error: 0.0008998705586733935
```



Discussion

The output plot for clean speech for the LMS algorithm sampled at 44.1kHz has a lot of samples and shows a relatively smooth curve of converging error. Larger window sizes show a smoother error reduction curve. Smaller window sizes show a sharper and faster convergence. This could be because in a sample of 16 values of speech signal sampled at 44.1kHz, there is very little variance between the samples. The output prediction will be very close to the training values and the error is smaller. This can be seen in figure 3 where the initial error is smaller than that of figure 1 and 2.

Introducing the non-linearity causes some very visible changes in the error convergence curves. I have applied both Sigmoid and Tanh non linearity and we can see that tanh performs better than the sigmoid activation function. Tanh is a logistic rescaled version of the Sigmoid activation function. The outputs of the sigmoid activation function are centered around 0.5 and that of tanh are centered around 0. It is well known that convergence is usually faster when the average of each input variable is centered around 0. Since our input speech signal lies between -1 and 1, the tanh activation function gives better results. Again we see faster convergence and lower errors with reducing window sizes due the reduced variance. Using larger window sizes has a smoothing effect on the error peaks and we therefore see fewer of these.

To better see the results, the mixed noise and clean speech signal was sampled at 8kHz. This also reduces the variability between adjacent samples. But overall we can see that images 10 and above follow the same trend as the clean speech error plots. We can see that adding noise means that the error doesn't converge as well as it did for the clean speech signal. But then again we have to remember that the number of samples here is also lower. We might have got better results for the same number of samples. Increasing window size again increases the time taken for convergence. Tanh activation function performs better than the Sigmoid activation function.

We also tend to see a convergence of error at the end of each speech segment. This seems to increase again when the speech picks back up again.

Another interesting thing about the error plots is that the sudden peaks in the error plots correspond to the peaks in the input audio signal. Speech signal is highly uncorrelated and unpredictable. Error plots for something as simple as a pure sine wave or strumming of a guitar would converge much better than that of speech just because there is more predictability between adjacent samples.

A simple LMS algorithm seems to produce extremely competent and in some cases better error convergence than when non-linearity is introduced. When non-linearity is introduced there are a lot of parameters that need to be altered including learning rate etc. LMS has less dependencies. There is also no black box here due to the transparency and the lack of hidden layers.