AUDIO RESTORATION OF DEGRADED AUDIO BY REMOVING CLICKS

Report for Module EEP55C22 Computational Methods

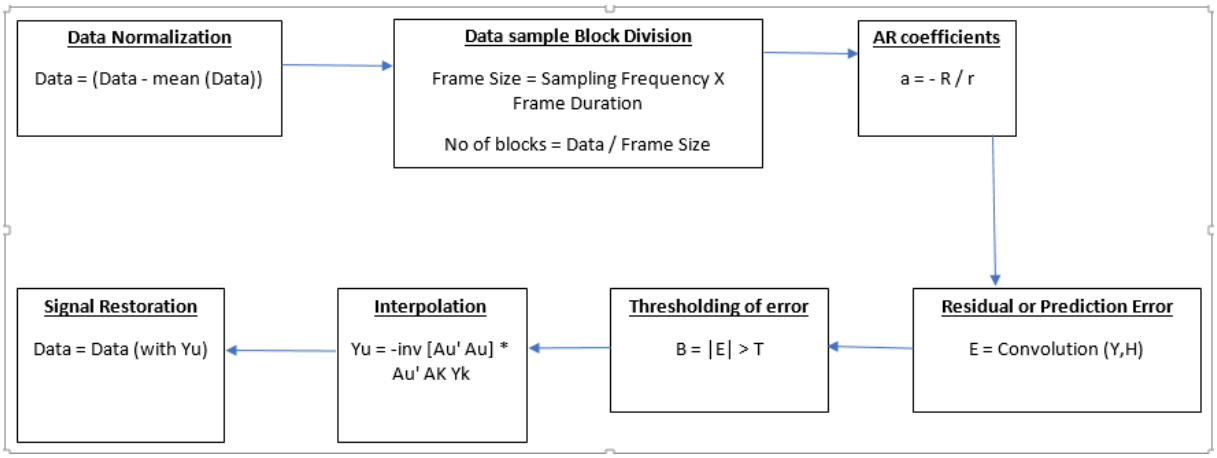
Abhishek Duttagupta, Student ID 22312353
Trinity College Dublin
aduttagu@tcd.ie

November 1, 2022

This report is submitted in part fulfilment for the assessment required in EEP55C22 Computational Methods. I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year. These are found in Parts II and III at http://www.tcd.ie/calendar.

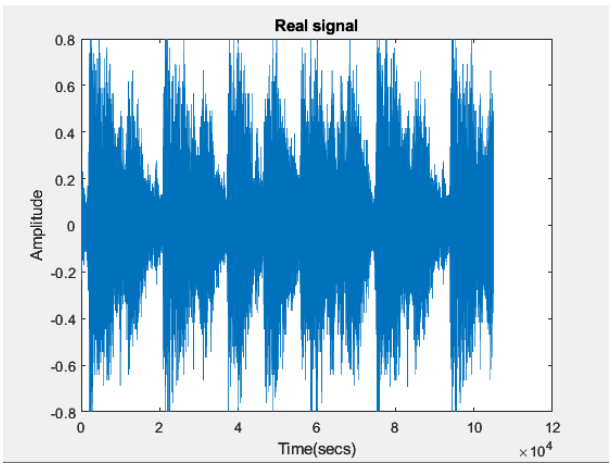
This report describes the algorithm and the MATLAB code designed for detection and removal of clicks in degraded audio tracks. The sample clean audio undertaken for this project is a ten seconds clip, which is then degraded artificially and restored again. The restored clip is then compared to the original clean clip and different performance analysis charts, graphs and tables are plotted.

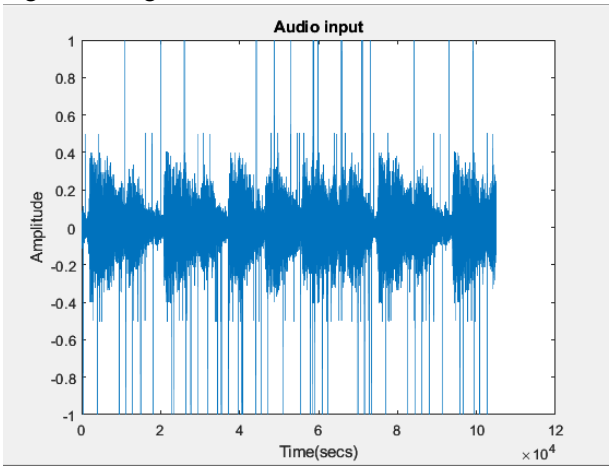
1 Algorithm, Block Diagram and brief description of AR process



1.1 Degradation of Real Audio

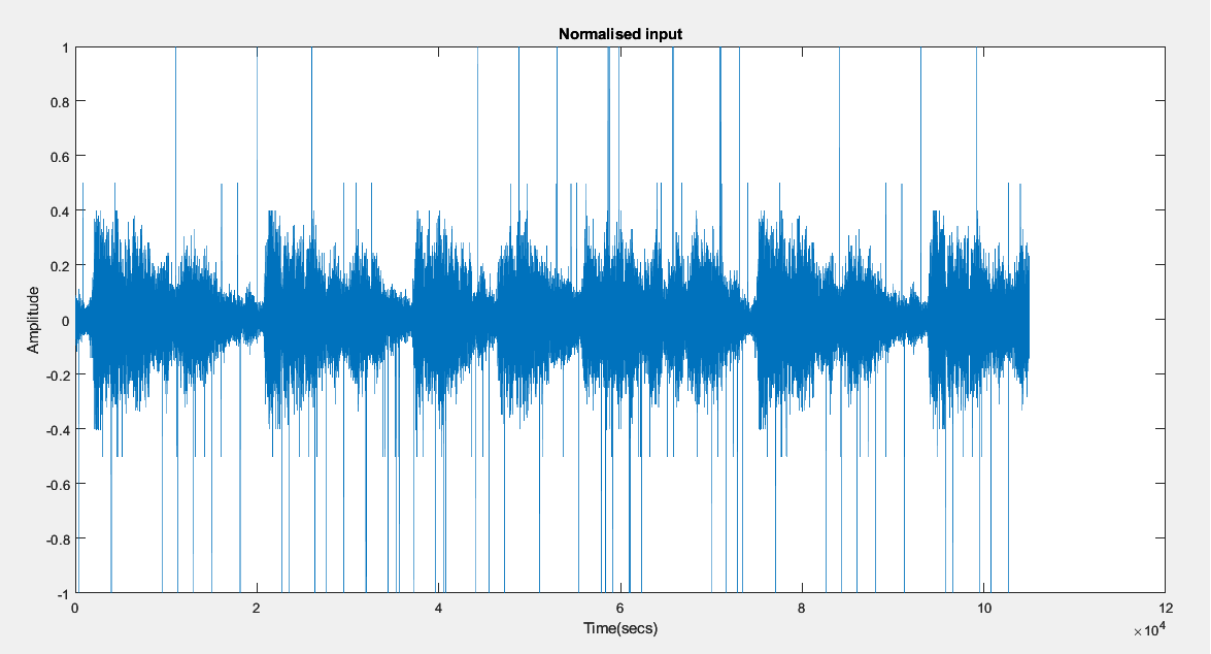
A real audio with clip length of 10 seconds is taken, which is then degraded artificially. The degrading process includes periodically adding random values to the audio. Below, on the left we see the Real signal and on the right the artificially degraded signal





1.2 Data Normalization

Data normalization is done by removing DC levels from the signal by subtracting the mean value from the data. The data is normalized so that we don't end up predicting dc level. Below is the plot of the degraded signal after Normalization.



1.3 Data block division

The complete data is divided into an equal number of blocks, so that all the other processes are processed block by block and then joined together to get the complete signal back. The block division is done based on the frame duration which is calculated from the sampling frequency and selected frame time. Variation in the frame time leads to distinct size of the blocks.

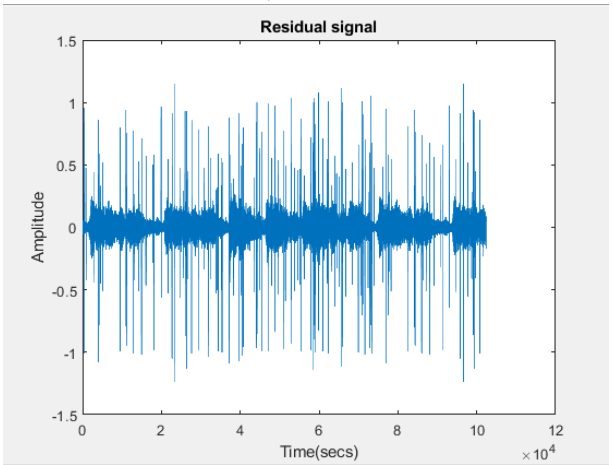
1.4 AR Coefficients

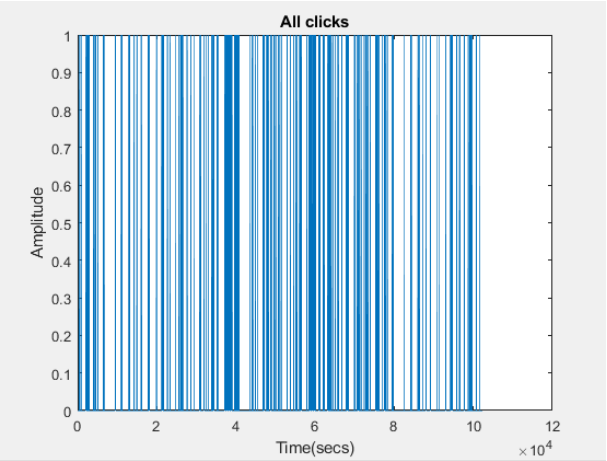
With the help of coefficient and previous samples at a position, we try to predict the next samples. The method of least squares is used to compute the error between the original and the restored signal over a range of data blocks in the audio, which helps to compute the AR coefficients using standard matrix solvers. This is usually summarised as a = -R / r.

1.5 Residual Calculation and error threshold

The residual or the prediction error is just the difference between the actual signal and the prediction.This is just a convolution of Y with a FIR filter H, which has the AR coefficients as its taps. Once the residuals are computed, they are accumulated and plotted. By looking at the graph, we try to find the threshold so that the error limit can be set and the signal can be interpolated based on that.Below in the left,is the plot of the accumulated residual from the degraded signal.In the

right, is the plot of clicks detected at various location.(Clicks is plotted from the degraded Matrix which makes all clicks 1, if more than the threshold error value).



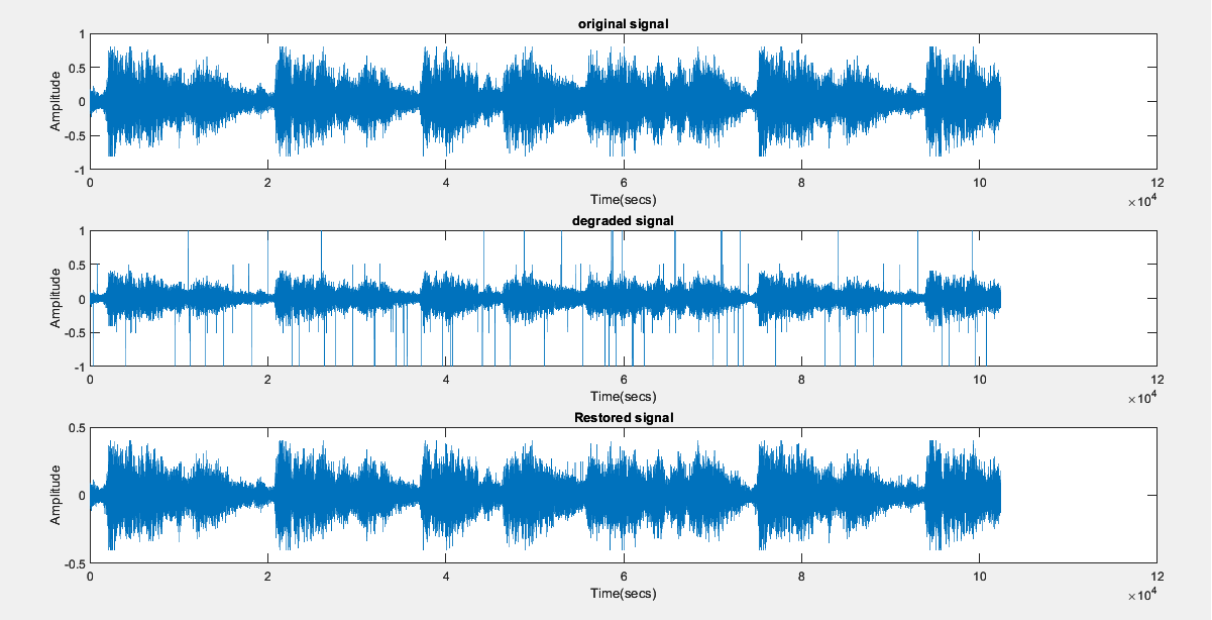


1.6 Interpolation

Interpolation is the method of computing the unknown values of signal after setting thresholds the prediction error. The whole process revolves around calculating the matrix A, which is the combination of known samples and missing samples.

1.7 Signal Restoration

After interpolation, the unknown values are plugged into the signal and the whole signal is reconstructed and restored, fresh from the clicks. Below is the plot comparing the real signal, degraded signal and the restored signal



2 Experiments and results

The whole AR process revolves around the following parameters, with change of each parameter leading to different MSE(mean square error). The MSE is calculated between the original clean signal and restored signal.

1. The Data block size
2. The model order
3. The error threshold

In the first table, we tabulate the MSE, keeping error threshold and block size constant and varying model order. We notice that the MSE increases till model order 10, and then started decreasing till 15, with again increasing again. So we select 15 as the best model order with all the model order approximately taking the same time.

Keeping Blocksize = 4096, Threshold = 0.25 constant and varying the Model Error to compute the results					
SR. NO	BLOCK SIZE	MODEL ORDER	THRESHOLD ERROR	MEAN SQUARE ERROR	TIME(TIC - TOC) - secs
1	4096	2	0.25	0.00011169	2.522
2	4096	3	0.25	0.00016021	2.478
3	4096	5	0.25	0.00019110	2.487
4	4096	15	0.25	0.00008226	2.513
5	4096	20	0.25	0.00016772	2.654
6	4096	50	0.25	0.00017019	3.013

In the second table, we tabulate the MSE, keeping error threshold and model order constant and varying block size. We notice that the MSE decreases with increase in block size, even though the time required for increases with increase in block size. Hence we proceed as 8192 with the block size.

Keeping Threshold Error = 0.25, Model Order = 15 constant and varying the Block size to compute the results					
SR. NO	BLOCK SIZE	MODEL ORDER	THRESHOLD ERROR	MEAN SQUARE ERROR	TIME(TIC - TOC) - secs
1	1024	15	0.25	3.3664E-04	0.843
2	2048	15	0.25	1.6831E-04	1.305
3	4096	15	0.25	8.2261E-05	2.513
4	6144	15	0.25	5.6101E-05	2.782
5	8192	15	0.25	3.9633E-05	3.703

In the third table, we tabulate the MSE, keeping block size and model order constant and varying threshold error. We notice that the MSE is most for 0.15, with 0.20 and 0.25 having the least MSE. 0.15 has the least MSE because, we end up cutting crucial audio signal part that again results to click. However, looking and the residual graph before, 0.25 seems the best suitable.

Keeping Blocksize = 8192, Model Order = 3 constant and varying the Threshold Error to compute the results					
SR. NO	BLOCK SIZE	MODEL ORDER	THRESHOLD ERROR	MEAN SQUARE ERROR	TIME(TIC - TOC) - secs
1	8192	15	0.15	3.9672E-05	3.925
2	8192	15	0.2	3.9616E-05	3.787
3	8192	15	0.25	3.9633E-05	3.703
4	8192	15	0.3	3.9641E-05	3.657
5	8192	15	0.4	3.9645E-05	3.645

3 Performance Analysis

Once the parameters are selected for the model, the performance of that system is analysed. Based on that, a confusion Matrix of the signal is created, wherein True Positives, True Negatives, False Positives and False Negatives are tabled to calculated the accuracy.

Model Order = 15

Block size = 8092

Threshold error = 0.25

Total number of Data points compared = 98304

Number of detected clicks = 272

Now passing the restored signal again through the Audio restoration process to check how many clicks were detected successfully, we can estimate the false positives or false negatives. Also passing the original audio, we can check if any original data is being considered as a click or not, which is a False negative. Number of clicks detected(in the above manner) = 134

Number of False positives or False Negatives = 134. Accuracy = (98304 - 134) / 98304 = 99.7 percentage.

4 Conclusions / Takeaways

The AR method really worked, as we were able to detect the clicks and remove them. The following are the takeaways or difficulties faced during the assignment

1. Using of functions are vital, initially I started the program without function, which leads to lot of computational matrix being 3 Dimensional and difficult to visualise in MATLAB platform.
2. While dividing the whole audio in number of blocks, certain number of end data points gets omitted, because of the use of floor function.
3. Perfection of selection of the error threshold, as sometimes it might lead to original sound data being chop off, which itself behaves as a noise.
4. Creation of confusion matrix for performance analysis in my methodology needed to rerun the restored version audio through the code again to check how many clicks were correctly detected (False Positive) and re-running the click audio through the code if any of original data is being considered as a click or not, which is a False - Negative.

5 Appendix - MATLAB CODE

5.1 AR COEFFICIENTS

Algorithm 1 AR coefficients

```
procedure AR COEFFS(r)                                          ▷ Making Matrix r
    for i = 1 :  modelorder do
        r(i) = sum(data((modelorder + 1):length(data)) .* data((modelorder + 1 - i) :
(length(data) - i)));
    end for
end procedure
procedure AR COEFFS(R)                                          ▷ Making Matrix R
    for i = 1 :  modelorder, J = 1 :  modelorder do
        R(i, j) = sum(data((modelorder + 1 - i) :  length(data) - i) .*
data((modelorder + 1 - j) :  (length(data) - j)));
    end for
end procedure
procedure AR COEFFS(a)                          ▷ coeffs = -inv(R) * transpose(r);
end procedure
```

5.2 AR RESIDUAL

Algorithm 2 Residual

procedure RESIDUAL(res) ▷ Computing Residual

$residual \leftarrow filter([1 coeffs], 1, data);$

end procedure

5.3 AR INTERPOLATION

```
% creation of A Matrix

    A = zeros([(fram_size - model_order) fram_size]);
    for i=1:fram_size - model_order
        int_coef=[flip(coeffs),1];
        A(i,i:length(int_coef)+i-1)=int_coef;
    end

% Creating Matrix Ak, Au out of A and yk.

    u_ind=find(thres_res==1);
    k_ind=find(thres_res==0);
    Ak=A(:,k_ind);
    Au=A(:,u_ind);
    yk=datablock(:,k_ind);

%unknown elements (yu) from matrix calculations
    yu = (- (Au)' * Au) \ ((Au)' * Ak * (yk)');
```

Bibliography

[1] SHAMAN, P., AND STINE, R. A. The bias of autoregressive coefficient estimators. *Journal of the American Statistical Association 83*, 403 (1988), 842–848.

[2] VASEGHI, S. V., AND RAYNER, P. Detection and suppression of impulsive noise in speech communication systems. *IEE Proceedings I (Communications, Speech and Vision) 137*, 1 (1990), 38–46.

[1] [2] [3] Katsenou Angeliki 55C22 Lectures and Labs