READ ME

ECG De-noising using Extended Kernel Recursive Least Squares Adaptive Filter Algorithm

ABOUT

The objective of this project is the implementation of Extended Kernel Recursive Least Squares Adaptive Filtering Algorithm for the filtering of five different noises that occur in ECG signals during data acquisition.

INTRODUCTION

There are five different types of noises that can corrupt an ECG (Electro-Cardio--Gram) signal – namely

- White Gaussian Noise (WN)
- Power Line Interference (PLI)
- Baseline Wander Noise (BWN)
- Electrode Movement (EMN)
- Muscle Artifacts (MAN)

There are various methods available to filter these noises and researchers have embarked on the quest for finding a filter that performs well over all the noises. One particular approach is based on the Recursive Least Square Method of the family of Adaptive filter algorithms. This MATLAB Project implements the same as an experimental verification of their research. (See reference)

Further improvements have been made in the RLS algorithm by implementing the Extended Kernel RLS algorithm. The EKRLS uses the knowledge of the state transition matrix of the system and exponential weighting factor to improve the performance of RLS filter. The EKRLS have better tracking performance and computational efficiency. For comparison, the two algorithms have been presented along with the commonly used Least Mean Square (LMS) algorithm. The pseudo codes of both RLS and EKRLS algorithms are provided at the end of this document.

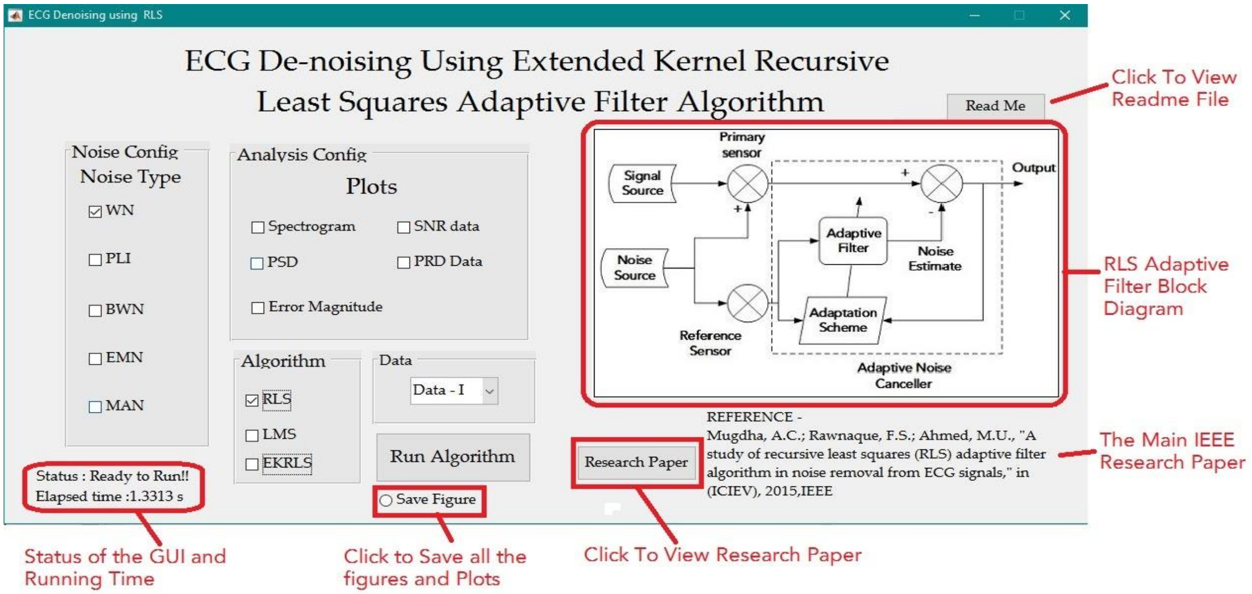
SETUP

- MATLAB 2013 or higher version installed in Windows 7 or higher
- Extract the Contents of the zip file
- Open the main.m MATLAB script file
- Run the Program and click “Add to path” when prompted

USAGE

- Select the Noise Type under the “Noise Config” Block
- Select the Required Plots under the “Analysis Config” Block
- Select the type of Filter Algorithm
- Select the Dataset under the “Data” Block
- Click “Save Figure” for all the plots to be saved in the folder names “Plot_Images”

- Click “Run Algorithm” Button.
  Note: Always select the algorithm(s) before clicking the Run button.
- Click the “Research Paper” Button to view the IEEE Research Paper
- Click the “Read Me” Button to view this read me file
- Once the program has finished its execution, the status and the run time will be displayed on the bottom left. Refer Fig.1



ECG Denoising using RLS

ECG De-noising Using Extended Kernel Recursive Least Squares Adaptive Filter Algorithm

Noise Config
Noise Type
☑ WN
□ PLI
□ BWN
□ EMN
□ MAN

Analysis Config
Plots
□ Spectrogram
□ SNR data
□ PSD
□ PRD Data
□ Error Magnitude

Algorithm
☑ RLS
□ LMS
□ EKRLS

Data
Data - I

Status : Ready to Run!!
Elapsed time :1.3313 s

○ Save Figure

REFERENCE -
Mugdha, A.C.; Rawnaque, F.S.; Ahmed, M.U., "A study of recursive least squares (RLS) adaptive filter algorithm in noise removal from ECG signals," in (ICIEV), 2015,IEEE

Click To View Readme File

RLS Adaptive Filter Block Diagram

The Main IEEE Research Paper

Status of the GUI and Running Time

Click to Save all the figures and Plots

Click To View Research Paper

Fig.1

DATASETS

The datasets were downloaded from Physionet ECG Database. Info files (.info) have been provided for all the various datasets downloaded from the given website for additional information regarding the datasets. All the required documents are in the folder names “Docs”. There are 3 datasets used this algorithm with increasing noise level for analysing the performance of the filters.

HELP

If you experience difficulties using this program, first make sure that the steps in Section 2 have been completed. This project has been licensed under GPL v3.0. You can use the code, modify it and improve freely.

Please direct questions/comments to:
Anand Krish
anandkrish894@outlook.com

REVISION VERSION 1.7

REFERENCE

[1] Mugdha, A.C.; Rawnaque, F.S.; Ahmed, M.U., "A study of recursive least squares (RLS) adaptive filter algorithm in noise removal from ECG signals," in (ICIEV), 2015,IEEE

APPENDIX

Algorithm Pseudo-Codes

$\boldsymbol{W}$	Weight vector
$\boldsymbol{P}$	Inverse Correlation Matrix
$\boldsymbol{A}$	State transition Matrix
$\boldsymbol{y}$	noise vector
$\boldsymbol{d}$	Input vector
$\boldsymbol{e}$	Error Vector
λ	Regularization parameter
k	Gain
β	Exponential Weighting factor
q	Bias for measurement disturbance

Algorithm 1: RLS Algorithm

```
Initialize $\boldsymbol{W} = 0$ and $\boldsymbol{P} = \lambda^{-1}\boldsymbol{I}$
Iterate for $i \geq 1$
    $\boldsymbol{r}(i) = \boldsymbol{W}^T\boldsymbol{y}$
    $\boldsymbol{P_i}(i) = \boldsymbol{P}(i)\boldsymbol{y}$
    $k = \frac{\boldsymbol{P_i}(i)}{\lambda + \boldsymbol{y}^T\boldsymbol{P_i}(i)}$
    $\boldsymbol{e}(i) = \boldsymbol{d}(i) - \boldsymbol{r}(i)$
    $\boldsymbol{W} = \boldsymbol{W}(i-1) + k\boldsymbol{e}(i)$
    $\boldsymbol{P}(i) = \frac{1}{\lambda}(\boldsymbol{P}(i-1) - k\boldsymbol{y}^T\boldsymbol{P}(i-1))$
```

Algorithm 2: EKRLS Algorithm

```
Initialize $\boldsymbol{W} = 0$ and $\boldsymbol{P} = \lambda^{-1}\boldsymbol{I}$
Iterate for $i \geq 1$
    $\boldsymbol{r}(i) = \boldsymbol{W}^T\boldsymbol{y}$
    $\boldsymbol{P_i}(i) = \boldsymbol{P}(i)\boldsymbol{y}$
    $k = \frac{\boldsymbol{A}\,\boldsymbol{P_i}(i)}{\beta + \lambda + \boldsymbol{y}^T\boldsymbol{P_i}(i)}$
    $\boldsymbol{e}(i) = \boldsymbol{d}(i) - \boldsymbol{r}(i)$
    $\boldsymbol{W} = \boldsymbol{A}\boldsymbol{W}(i-1) + k\boldsymbol{e}(i)$
    $\boldsymbol{P}(i) = \frac{1}{\lambda}\boldsymbol{A}\big(\boldsymbol{P}(i-1) - k\boldsymbol{y}^T\boldsymbol{P}(i-1)\big)\boldsymbol{A}^T + \beta q\boldsymbol{I}$
```