Detection of PAF using ECGs

JOSHUA PETER EBENEZER – 15EC10023

DIGITAL SIGNAL PROCESSING PROJECT

ABSTRACT

This report presents a scheme to differentiate between the ECG (Electrocardiographs) signals of patients having healthy hearts and patients showing symptoms of paroxysmal atrial fibrillation (PAF). ECGs of patients with PAF were found to have higher energy content in the lowpass region (less than 1 Hz), than the ECGs of healthy patients. The auto-correlation of the ECG signals for patients with PAF, for a spacing of 1s, was found to be greater than that of healthy patients. The product of these two quantities was defined as a feature, and a maximum likelihood classifier based on this feature gave 100% precision and 100% recall on test data.

INTRODUCTION

The task is to find a feature or features that can differentiate between ECG (Electrocardiographs) signals of patients having healthy hearts and patients showing symptoms of paroxysmal atrial fibrillation (PAF). Atrial fibrillation occurs when the atria, or upper chambers of the heart, lose their normal rhythm and beat chaotically. As the population ages, the prevalence is expected to rise; currently approximately 6% of the US population over the age of 65 are diagnosed with this arrhythmia. The development of accurate predictors of the acute onset of PAF is clinically important because of the increasing possibility of electrically stabilizing and preventing the onset of atrial arrhythmias with different atrial pacing techniques.

DATABASE

The database of two-channel ECG recordings was created for use in the Computers in Cardiology Challenge 2001, and is available in PhysioNet. The learning set that was used by the author consists of 24 records. Each record has two signals, both taken of the same patient at the same time with two different electrodes. Each signal is of 5 minutes' duration and sampled at 128 Hz. For training, 14 records were used. 7 records were those of normal patients (record name prefixed with 'n'), and the other 7 were of patients having PAF (prefixed with 'p'). 6 records (3 normal and 3 abnormal) were used for validation, and 4 (2 normal and 2 abnormal) were used for testing. The files were downloaded using the PhsyioNet ATM as .mat and .info files.

CHARACTERISTICS OF THE DATA

The signals were first normalized with respect to total energy (so that all signals have unit total energy). A normal ECG signal, titled 'n01cm.mat' in the database, is shown in Fig. 1 (taken from training

set), before normalization. An ECG of a patient with PAF, titled ‘p14cm.mat’ in the database, is shown in Fig. 2 before normalization.

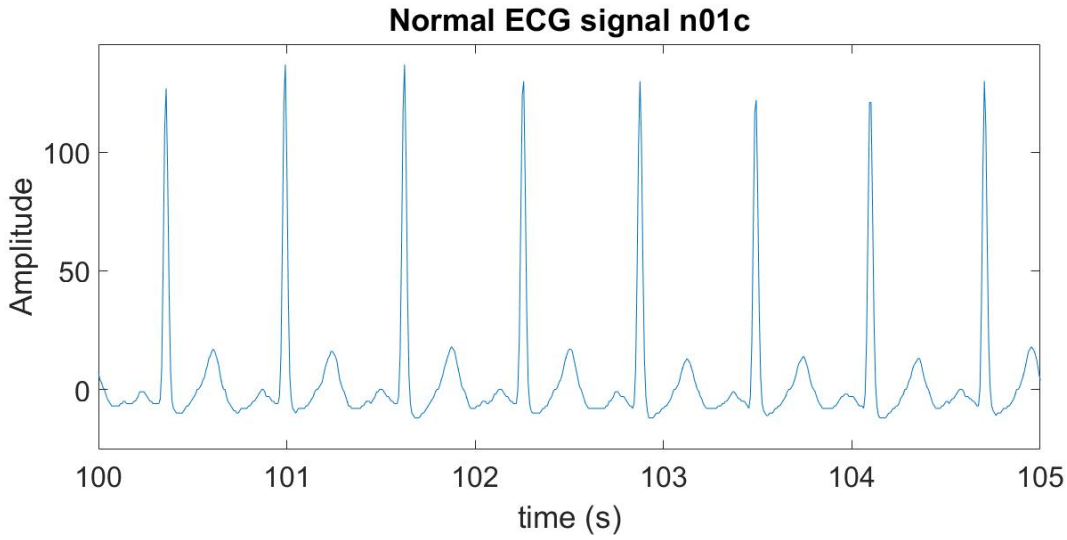

Fig. 1 *Normal ECG signal (time domain)*

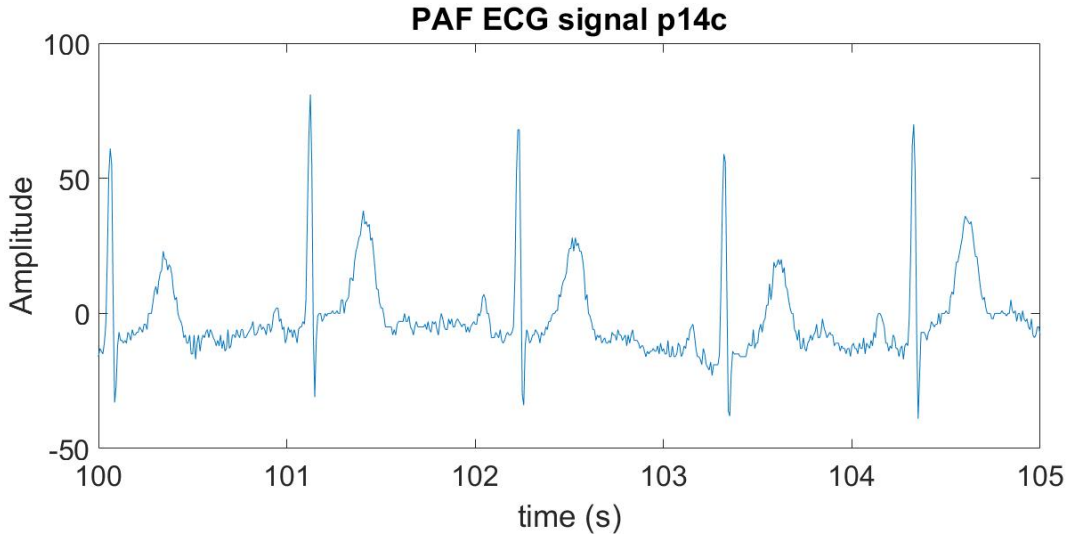

Fig. 2 *ECG signal of patient with PAF (p14c)*

The FFTs of a few normal signals (after energy normalization) are shown in Fig. 3. FFTs of a few abnormal signals (also after energy normalization) are shown in Fig. 4. The spread of the data might be misleading. Abnormal signals actually have a larger peak at 0 Hz than normal signals, hence the spectra near 0 Hz looks smaller in amplitude in comparison to that of normal signals, but it is actually greater. Hence, the summed energy of the lowpass portion of abnormal signals is greater than that of normal signals. This is because the fluctuations in PAF cause a larger, non-zero DC average. Also, while normal signals do not have an envelope, most PAF signals have a very low frequency slowly varying envelope.

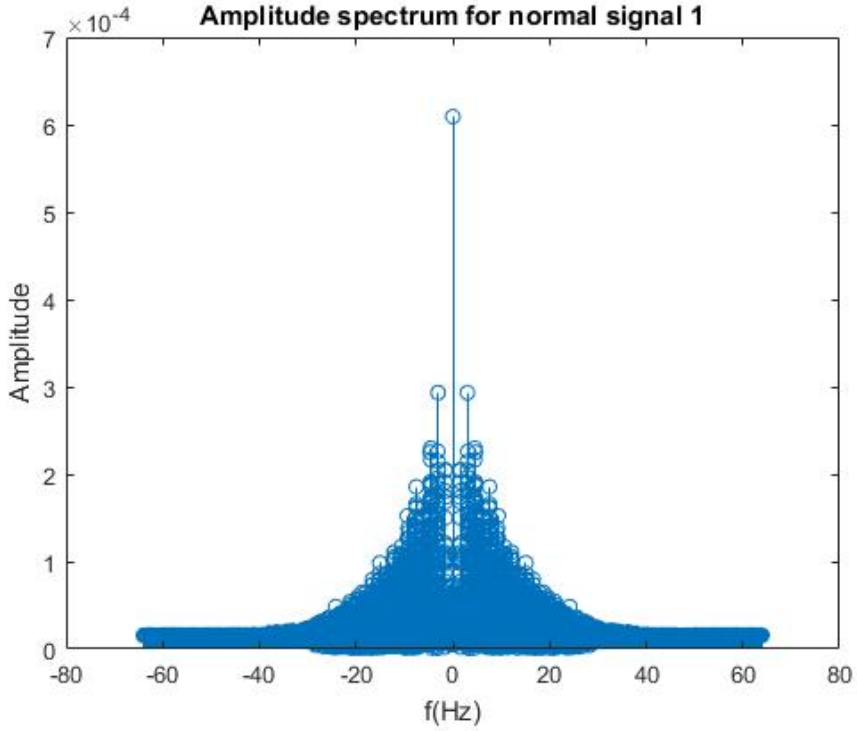

Fig. 3 *Spectrum of normal ECG signal*

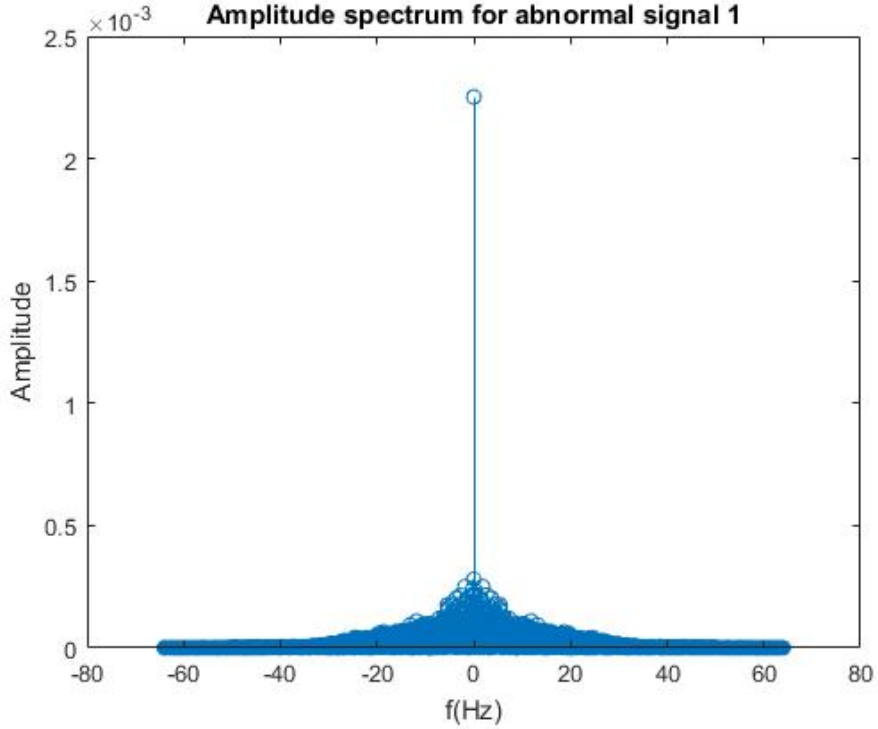

Fig. 4 *Spectrum of abnormal ECG signal*

The absolute auto-correlation of normal signals for a spacing of 1s (around the average duration of a heart beat) is lesser than that of PAF signals. This can also be attributed to the variations in PAF, fluctuating away from baseline with greater amplitudes. Table 1 and 2 show the parameters for data in the training set. Auto-correlation is displayed for signal 2 of each normal record, and signal 1 of each abnormal record. Energy is the average of the signals in each record (after filtering).

Table 1 *Auto-correlation and lowpass (<1 Hz) energy for normal ECGs in training*

RECORD LABEL	$R_{xx}(10)$	LOWPASS ENERGY
'N01CM.MAT'	0.000370941696483284	0.109298189625764
'N02CM.MAT'	-0.128461888617335	0.723823502299284
'N03CM.MAT'	0.0554480814673563	0.186105813334087
'N04CM.MAT'	0.0290757474668627	0.0877805656872497
'N05CM.MAT'	0.0262571598116157	0.00665922490603559
'N16CM.MAT'	0.00426915509837172	0.0488389828247902

Table 2 *Auto-correlation and lowpass(<1 Hz) energy for abnormal ECGs in training*

RECORD LABEL	$R_{xx}(10)$	LOWPASS ENERGY
'P14CM.MAT'	0.612175851293025	0.970108654392964
'P16CM.MAT'	0.00999296613949006	0.432943239808761
'P18CM.MAT'	0.553005552553372	1.80008845747379
'P20CM.MAT'	0.0351482782748674	0.381312909556024
'P22CM.MAT'	0.334380459072233	0.785214686372720
'P24CM.MAT'	0.360541346407814	0.517542148594771

The mean absolute auto-correlation for normal signals in training is **0.0370**, and for abnormal signals is **0.2977**. The energy of the filtered normal signals is **0.3263**, and that of filtered abnormal signals is **0.9283**. Hence the product of these two averages for normal signals is much lower than that of abnormal signals.

FILTER USED

The data in the training set was used to find thresholds for energy and auto-correlationThe signals were passed through a Butterworth filter with lowpass cutoff frequency 1 Hz (0.005π in digital domain), and an order of 4. The frequency response of the filter is shown in Fig. 5.

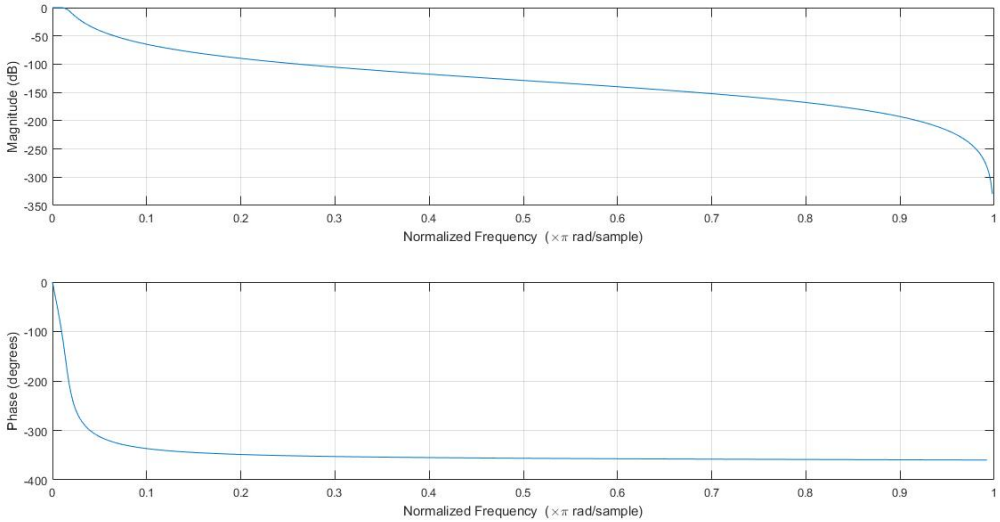

Fig. 5 *Frequency response (magnitude and phase) of the filter used*

Hence, the filtered signal has energy content mostly only in the lowpass region. The energies of the filtered signal were different for normal and abnormal signals, and are tabulated in Tables 1 and 2.

VALIDATION AND TESTING

The two products of the mean energy and mean autocorrelation (for normal and abnormal signals in the training set) were weighted by a factor between 0.6 and 1. Data in the validation set were classified based on their relative distance from the weighted products. A signal was classified as normal if the product of the energy of the lowpass filtered signal and the autocorrelation of the signal was closer to the corresponding weighted product of the normal training set data than the weighted product of the abnormal data in the training set. All the factors gave 100% accuracy in the validation set, and the factor of unity was chosen finally. Hence the means were used as the thresholds, forming a maximum likelihood classifier.

For testing, the auto-correlation of the signal for a spacing of 1s was found and stored. Then the signal was filtered through the same lowpass filter described earlier. The energy of the filtered signal was found. The product of the energy and the auto-correlation formed the comparable quantity. The product space was divided by the maximum likelihood classifier into two regions. If the product of energy and auto-correlation the test signal was closer to the mean of the product of the energy and auto-correlation of the normal training data, the test signal was classified as normal. If the product was closer to the mean of the product of the energy and auto-correlation of the abnormal training data, the test signal was classified as abnormal. The product space is one dimensional.

The precision and recall were both 100% for the testing set.

CONCLUSION

The report shows that the product of the energy and the autocorrelation may be an important consideration while differentiating between PAF and normal signals.

MATLAB® CODE

```
clear;
clc;
close all;

fnames=dir('train/*.mat');
normal = zeros(7,2,38400);
arrhy = zeros(7,2,38400);

val_normal = zeros(3,2,38400);
val_arrhy = zeros(3,2,38400);

test_normal = zeros(4,2,38400);
test_arrhy = zeros(4,2,38400);

Fs = 128;
% number of sampling instances
N = 38400;
% frequency base
f = (-N/2:N/2-1)*Fs/N;
% time base
t = 0:1/Fs:5*60-1/Fs;

%% Read the data and normalize it
for i=1:14
    file=fullfile('train',fnames(i).name);
    s = load(file);
    if (i<=7)
        normal(i,:,:)=s.val;
        % Find energy of the signal and normalize
        energy1 = sum(normal(i,1,:).^2);
        energy2 = sum(normal(i,2,:).^2);
        normal(i,1,:)=normal(i,1,:)/sqrt(energy1);
        normal(i,2,:)=normal(i,2,:)/sqrt(energy2);

    else
        arrhy(i-7,:,:)=s.val;
        % Find energy of the signal and normalize
        energy1 = sum(arrhy(i-7,1,:).^2);
        energy2 = sum(arrhy(i-7,2,:).^2);
        arrhy(i-7,1,:)=arrhy(i-7,1,:)/sqrt(energy1);
        arrhy(i-7,2,:)=arrhy(i-7,2,:)/sqrt(energy2);
    end

end
```

```
n = 4;
Wn = 1/(Fs/2);
% Zero-Pole-Gain design
[z,p,k] = butter(n,Wn,'low');
sos = zp2sos(z,p,k);
enorm =zeros(7,1);
earrhy = zeros(7,1);

ncor1 = zeros(7,1);
ncor2 = zeros(7,1);
acor1 = zeros(7,1);
acor2 = zeros(7,1);

for i=1:14
    if (i<=7)
        s1=normal(i,1,:);
        s2=normal(i,2,:);

        y1 = sosfilt(sos,s1);
        y2 = sosfilt(sos,s2);

        enorm(i) = enorm(i)+sum(y1.^2);
        enorm(i) = enorm(i)+sum(y2.^2);
        ncor1(i) = sum(s1(1,1,1:end-128).*s1(1,1,129:end));
        ncor2(i) = sum(s2(1,1,1:end-128).*s2(1,1,129:end));

    else
        s1=arrhy(i-7,1,:);
        s2=arrhy(i-7,2,:);

        y1 = sosfilt(sos,s1);
        y2 = sosfilt(sos,s2);

        earrhy(i-7) = earrhy(i-7)+sum(y1.^2);
        earrhy(i-7) = earrhy(i-7)+sum(y2.^2);
        acor1(i-7) = sum(s1(1,1,1:end-128).*s1(1,1,129:end));
        acor2(i-7) = sum(s2(1,1,1:end-128).*s2(1,1,129:end));

    end
end

%% ------------------------------ Validation ------------------------- %%
val_names=dir('val/*.mat');
norm_thresh = zeros(4,1);
arrhy_thresh = zeros(4,1);
confidence = zeros(4,1);

norm_measure = mean(enorm)*(abs(mean(ncor1))+abs(mean(ncor2)))/2;
abnorm_measure = mean(earrhy)*(abs(mean(acor1))+abs(mean(acor2)))/2;

for j=1:4
    weight = 0.6+j*0.1;
    norm_thresh(j) = norm_measure*weight;
    arrhy_thresh(j) = abnorm_measure*weight;
```

```
    for i=1:6
        file=fullfile('val',val_names(i).name);
        s = load(file);

        if (i<=3)
            val_normal(i,:,:)=s.val;
            % Find energy of the signal and normalize
            [measure1,measure2]=measure(val_normal,i);

            norm_true1 = abs(measure1-norm_thresh(j))<abs(measure1-
arrhy_thresh(j));
            norm_true2 = abs(measure2-norm_thresh(j))<abs(measure2-
arrhy_thresh(j));
            if ( norm_true1 && norm_true2)
                confidence(j)=confidence(j)+2;
            elseif (~norm_true1 && ~norm_true2)
                confidence(j)=confidence(j)-2;
            end
        else
            val_arrhy(i-3,:,:)=s.val;
            % Find energy of the signal and normalize
            [measure1,measure2]=measure(val_arrhy,i-3);

            abnorm_true1 = abs(measure1-norm_thresh(j))>abs(measure1-
arrhy_thresh(j));
            abnorm_true2 = abs(measure2-norm_thresh(j))>abs(measure2-
arrhy_thresh(j));

            if (abnorm_true1 && abnorm_true2)
                confidence(j)=confidence(j)+2;
            elseif (~abnorm_true1 && ~abnorm_true2)
                confidence(j)=confidence(j)-2;
            end

        end
    end
end

max_chk=fliplr(confidence);
[~,I] = max(max_chk);
weight = 0.6+I*0.1;
fin_norm_thresh = norm_measure*weight;
fin_arrhy_thresh = abnorm_measure*weight;


%% ------------------------------- Testing ---------------------------- %%
test_names=dir('test/*.mat');
tp = 0;
tn = 0;
fp=0;
fn=0;

for i=1:4
    file=fullfile('test',test_names(i).name);
```

```
    s = load(file);

    if (i<=2)
        test_normal(i,:,:)=s.val;
        % Find energy of the signal and normalize

        [measure1,measure2]=measure(test_normal,i);

        norm_true1 = abs(measure1-fin_norm_thresh)<abs(measure1-
fin_arrhy_thresh);
        norm_true2 = abs(measure2-fin_norm_thresh)<abs(measure2-
fin_arrhy_thresh);
        if (norm_true1)
            tp=tp+1;
        else
            fn=fn+1;
        end
        if (norm_true2)
            tp=tp+1;
        else
            fn=fn+1;
        end
    else
        test_arrhy(i-2,:,:)=s.val;

        [measure1,measure2]=measure(test_arrhy,i-2);

        abnorm_true1 = abs(measure1-fin_norm_thresh)>abs(measure1-
fin_arrhy_thresh);
        abnorm_true2 = abs(measure1-fin_norm_thresh)>abs(measure1-
fin_arrhy_thresh);

        if (abnorm_true1)
            tn=tn+1;
        else
            fp=fp+1;
        end
        if (abnorm_true2)
            tn=tn+1;
        else
            fp=fp+1;
        end

    end
end
precision=tp/(tp+fn)*100;
recall=tn/(tn+fp)*100;

function [measure1,measure2]=measure(input,i)


        energy1 = sum(input(i,1,:).^2);
        energy2 = sum(input(i,2,:).^2);
        input(i,1,:)=input(i,1,:)/sqrt(energy1);
        input(i,2,:)=input(i,2,:)/sqrt(energy2);
```

```
        y1 = sosfilt(sos,input(i,1,:));
        y2 = sosfilt(sos,input(i,2,:));

        energy1 = sum(y1.^2);
        energy2 = sum(y2.^2);
        test_acor1 = sum(input(i,1,1:end-128).*input(i,1,129:end));
        test_acor2 = sum(input(i,2,1:end-128).*input(i,2,129:end));

        measure1=energy1*abs(test_acor1);
        measure2=energy2*abs(test_acor2);

end
```