Mechanism for electromyography (EMG) signals recognition for human gait analysis

Author: Ivan Mitzev
Pattern Recognition, ECE 8443
Spring 2009

1. Abstract

Electromyography (EMG) signal recognition is complex process due to variety and non consistency of the EMG signals. This project is intended to develop a reliable and easy to implement method for EMG signals recognition for signals obtained during human gait. Recognition techniques are based on simple Mahalanobis distance measurement and Linear Discriminate Analysis (LNA) – both class dependant and class independent. Developed method uses five features characterizing EMG signals. Achieved error rate vary from 0% to 11.1% depending on investigated signal.

2. Introduction

EMG classification is widely used in electric control of mechanically developed prosthesis, robots development, clinical application etc. It has been evaluated for years and varieties of algorithms are developed to serve different goals. Focus of this project is to develop an easy to implement and fast to execute algorithm for classifying signals used for human gait analysis.
Laboratories investigated human gait collect all variety of EMG signals in binary format and store them on storage devices. As different people work on same acquisition system and store data in different folders, it is difficult to organize all of the scanned information.
Another obstacle for organizing is the variety of used file formats- C3D, CAMARC, TXT, CSV, among others. Most of the time, researches have to remember which file what kind of information contains. There is documentation of how such information has been collected, but it is tedious process to keep an eye on all the documentation surrounding acquiring process.
Current project easily solve these problems by using simple classification algorithm, which could be used as base for developing an EMG signal browser. Such browser will help to browse the whole storage device and find signals/files with similar to certain (defined earlier) type of EMG signal.

3. Proposed methods

Regarding the EMG classifier, there are several different approaches used in different systems. Among them, Bayes classifier, nonlinear discriminate analysis, principal component analysis plus linear discriminant analysis (PCA + LDA); *k*- nearest neighbors; as well some more advanced techniques such as fuzzy systems and artificial neural networks.

Signals used in this project are obtained from the following muscles: Rectus Femoris, Vastus Lateralis, Medial Hamstrings, Lateral Hamstrings, Adductors, Tibialis Anterior, Gastrocnemius, Gluteus Medialis, Gluteus Maximus and are downloaded from
ftp://ftp.emgsrus.com/sample/emgdata

Presented work is used five features extracted from row EMG signals. These features are *x1* = Zero Crosses, *x2* = Standard Deviation of the signal (STD), x3 = Pathway function, *x4* = Form Factor (FF) and *x5* = Area under the curve of the normalized signal. We are looking for parameters, which are stable and do not change much for particular muscle, but in the same time they differ from the same parameter values for another muscle.
Zero Crosses feature measure how many times the row signal crosses the zero axes. Pathway value is the "circumference" of the signal and determines the length of a signal; it is finding the straight line equal in length to the signal's curve. Standard deviation uses the following formula:

$$STD = \sqrt{\frac{\overline{x - \bar{x}}}{n-1}}$$, where

x represents EMG signal;
$\overline{x}$ represents the mean value of the signal
n is the number of scanned samples

Normalization of the signal is done using the following formula:

$$x[i] = \frac{x[i] - \overline{x}}{STD}$$, where

x[i] is current sample of EMG signal
$\overline{x}$ represents the mean value of the signal
STD is standard deviation of the signal.

To obtain the area under the curve of the signal, current project uses trapezoid rule. Form Factor is another unique characteristic, which initially has been developed for use with EEG signals, but later it become popular as good overall characteristic for EMG signals. Form factor is calculated using the following formula:

$FF = (\sigma_{x''}/\sigma_{x'})/(\sigma_{x'}/\sigma_x)$, where

σ_x is the variance of the signal and x' and x'' are the first and second derivate of the signal, respectively.

Algorithm for classification is trained with nine signals described above. Newly investigated signal will be classified after extracting the five features described above and by finding the Mahalanobis distance to all of the distributions. The smallest distance will point to the group of which the investigated signal belongs. If smallest distance is bigger than 13.82 (from chi-square table for p = 0.001) then newly arrived signal will not be classified. This project uses Linear Discriminate Analysis – both class dependant and class independent to clarify the solution. As seen from the results, LDA does not help much to improve the classification error. Classification error is obtained after taking training data from certain group and trying to classify them using proposed algorithm. All wrongly classified points are counted and their number is divided by nine (the size of the training set).

4. Results

Initial development of the project started with investigating x1/x2 parameters for three groups of muscles shown on Fig. 1- EMG1 (Rectus Femoris) drawn in red, EMG2 (Vastus Medialis) drawn in green and EMG2 (Vastus Lateralis) drawn in blue.

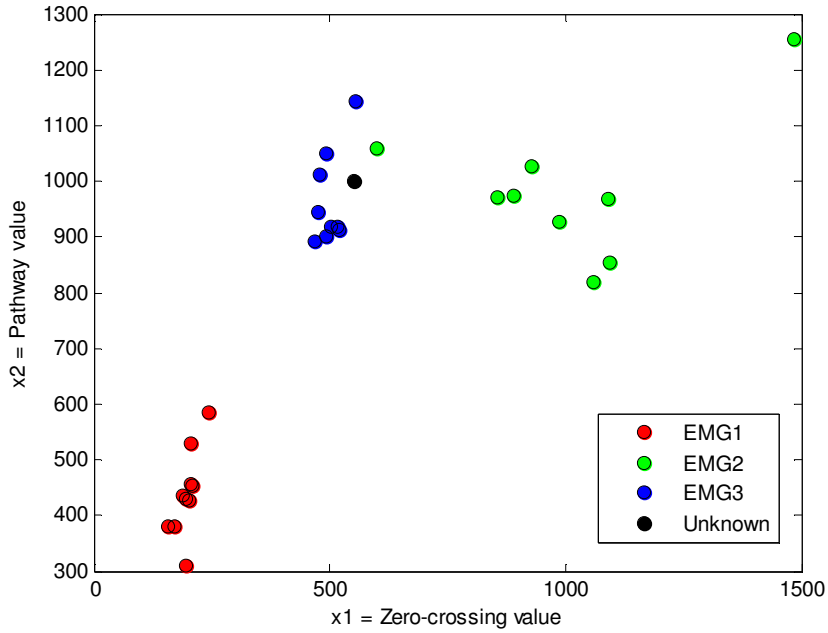

Fig. 1 Represent x1/x2 distribution for EMG1, EMG2 and EMG3 training set of signals.

As these three muscles are the only classified groups, it seems easy and clear to distinguish between them. In real world more signals are needed. Adding more muscles groups to the recognition system make our task more complicated. As shown on Fig. 2 many of investigated signals are overlapping each other.

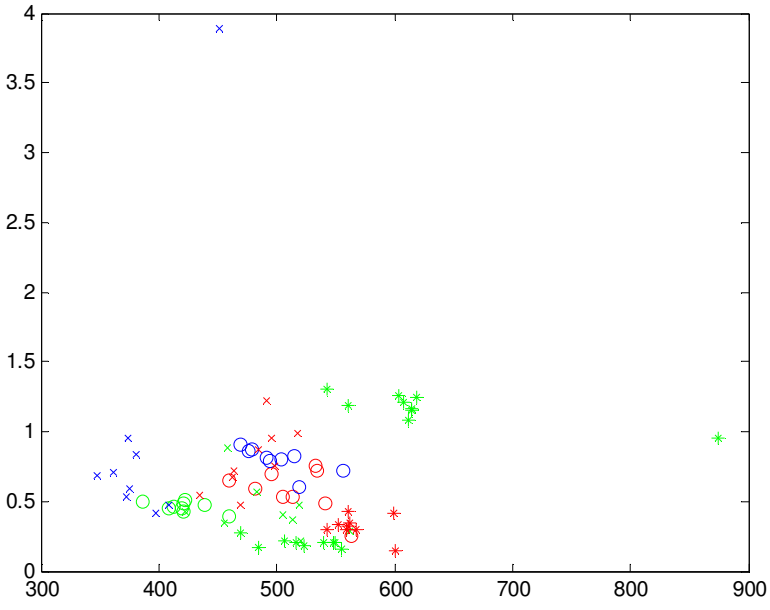

Fig. 2 Represent x1/x2 distribution for EMG1 through EMG9 training set of signals.

Using only two classification parameters produces errors which are in the range of 0.0 to 60% (Table 1), which is unacceptable. Introducing LDA may help to transform investigated points to the best differentiating space, but the results (Table 1) do not confirm that proposal and makes LDA not suitable for this task.

Signal	Error rate (simple Mahalonobis distance) [%]	Error rate (LDA) [%]	Error rate (LDA2) [%]
EMG1	55.6	55.6	55.6
EMG2	33.3	33.3	33.3
EMG3	55.6	55.6	55.6
EMG4	44.4	44.4	44.4
EMG5	55.6	55.6	55.6
EMG6	11.1	55.6	55.6
EMG7	11.1	11.1	11.1
EMG8	0.0	22.2	22.2
EMG9	22.2	22.2	22.2

Table 1. Error rates from recognition, using two factors (x1 = Zero Crosses, x2 = STD)

Another approach to make decision more precise is to increase the number of investigated parameters. Let's add one more parameter to the distribution picture: x3 = Pathway function of normalized signal. Fig. 3 shows the distribution for x1/x2/x3 for nine of the training EMG signals. Errors obtained from that step are clearly smaller than error obtained from x1/x2 distribution approach (Table 2).

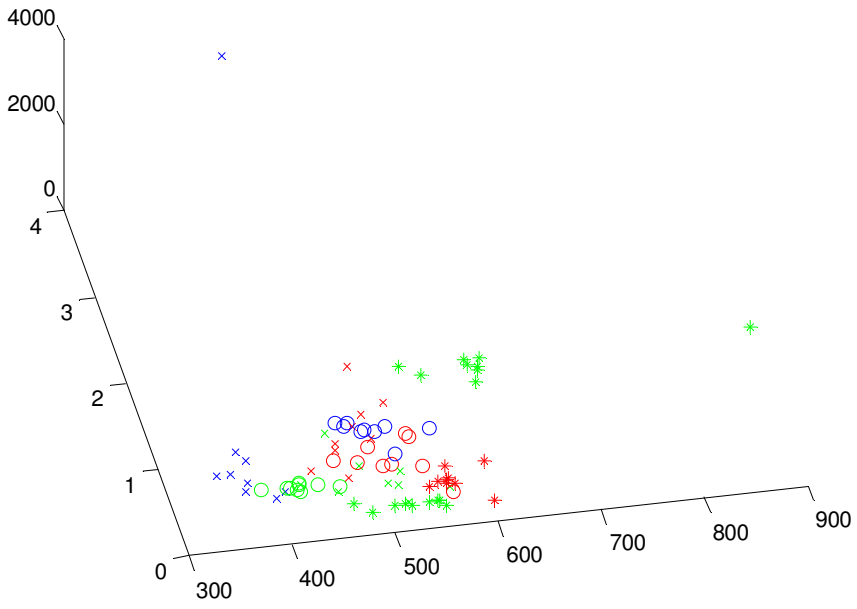

Fig. 3 Represent x1/x2/x3 distribution for EMG1 through EMG9 training set of signals.

Signal	Error rate (simple Mahalonobis distance) [%]	Error rate (LDA) [%]	Error rate (LDA2) [%]
EMG1	33.3	55.6	55.6
EMG2	22.2	22.2	22.2
EMG3	33.3	55.6	55.6
EMG4	44.4	55.6	55.6
EMG5	11.1	55.6	55.6
EMG6	0.0	22.2	22.2

EMG7	0.0	44.4	44.4
EMG8	0.0	11.1	11.1
EMG9	0.0	33.3	33.3

Table 2. Error rates using three factors (x1 = Zero Crosses, x2 = STD, and x3 = Pathway function)

As seen from Table 2, the error rates are still relatively high and do not give reliable solution. Increasing the number of compared features certainly gives better results; therefore we will try to introduce another independent parameter, which characterize the EMG signal. This parameter is x4 = Form Factor and Table 3 illustrate errors obtained after recognition with four features. Error is still relatively high and not suitable for practical use. Increasing number of parameters will decrease the performance in terms of calculation time, but if the parameter is simple to calculate and decrease of error is sensitive, it is worthy to include one more parameter into the estimation process. This parameter can be *x5* = Area under the curve of the normalized signal. Table 4 shows the results of this improvement. It is clearly visible that with five features recognition process is much more reliable and such system could be used for practical purposes.

Signal	Error rate (simple Mahalonobis distance) [%]	Error rate (LDA) [%]	Error rate (LDA2) [%]
EMG1	33.3	77.8	77.8
EMG2	11.1	66.7	66.7
EMG3	33.3	66.7	66.7
EMG4	11.1	44.4	44.4
EMG5	11.1	66.7	66.7
EMG6	0.0	11.1	11.1
EMG7	0.0	44.4	44.4
EMG8	0.0	11.1	11.1
EMG9	0.0	22.2	22.2

Table 3. Error rates from recognition, using four factors (x1 = Zero Crosses, x2 = STD, x3 = Pathway function, x4 = Form factor)

Signal	Error rate (simple Mahalonobis distance) [%]	Error rate (LDA) [%]	Error rate (LDA2) [%]
EMG1	0.0	66.7	66.7
EMG2	11.1	55.6	55.6
EMG3	0.0	44.4	44.4
EMG4	0.0	11.1	11.1
EMG5	11.1	11.1	11.1
EMG6	0.0	0.0	0.0
EMG7	0.0	22.2	22.2
EMG8	0.0	11.1	11.1
EMG9	0.0	0.0	0.0

Table 4. Error rates from recognition, using five factors (x1 = Zero Crosses, x2 = STD, x3 = Pathway function, x4 = Form factor, x5 = Normalized area)

5. Conclusions

a. EMG signals from muscles exercised during human gait, could be classified reliably using five parameters characterizing the row EMG signal: Zero crosses, STD, Form Factor, Pathway function, Area under the curve of the normalized signal

b. Linear discriminate analyses are not universal tool for developing better recognition systems. They could help to improve the performance of pattern recognition systems in some particular cases and certain signals features.
c. Increasing number of classification features brings better results and help to reduce the classification error rate.

References

1. Yi-Hung Liu*, Member, IEEE*, Han-Pang Huang*, Member, IEEE*, and Chang-Hsin Weng, "Recognition of Electromyographic Signals Using Cascaded Kernel Learning Machine", IEEE/ASME TRANSACTIONS ON MECHATRONICS, VOL. 12, NO. 3, JUNE 2007
2. Joseph Picone, Pattern Recognition Lectures, Mississippi State University, Spring 2009
3. Richard O. Duda, Peter E. Hart, and David G. Stork, "Pattern Classification"

APENDIX A (Source code used in the project)

EMG_classificator.m

```
% Project I, Pattern Recognition, Spring 2009
%
% This project is intended to recognize a newly arrived EMG signal to which
% of the previously recognized set belongs
%
function index = EMG_classificator(e)

%Number of samples :    2400
%Scan frequency:       800Hz
%x- Zerro crosses
%y- STD
%z- Pathway
%f- FF
%g- Normalized area

%EMG9
x(:,9) = [507; 549; 540; 555; 548; 523; 469; 517; 484];
y(:,9) = [0.218970; 0.203873; 0.206227; 0.156961; 0.201815; 0.186787; 0.275337; 0.207781;
0.166168];
z(:,9) = [194.855554; 212.655886; 210.698086; 177.522100; 200.195273; 215.120384;
216.478495; 191.049127; 193.624315];
f(:,9) = [1.949065; 1.888338; 1.764148; 1.856463; 1.844574; 1.672533; 1.924361; 2.027997;
1.874117];
g(:,9) = [1.816603; 1.951123; 1.934071; 2.025813; 1.911486; 2.078552; 1.681115; 1.889396;
2.323228];

%EMG8
x(:,8) = [543; 604; 615; 612; 607; 619; 561; 615; 874];
y(:,8) = [1.303635; 1.257568; 1.167121; 1.086307; 1.214580; 1.246938; 1.192898; 1.155678;
0.955456];
```

```
z(:,8) = [1128.092795; 1205.448441; 1201.459326; 1021.559444; 1163.191022; 1244.015533;
1090.350515; 1141.269490; 1841.943862];
f(:,8) = [1.575190; 1.556878; 1.484764; 1.529495; 1.420569; 1.603564; 1.566915; 1.537945;
1.258479];
g(:,8) = [1.539465; 1.515233; 1.529606; 1.457414; 1.453835; 1.572031; 1.464084; 1.492975;
2.134371];

%EMG7
x(:,7) = [567; 542; 562; 559; 561; 552; 599; 560; 601];
y(:,7) = [0.297111; 0.297484; 0.341431; 0.302723; 0.425680; 0.330622; 0.420241; 0.326378;
0.142905];
z(:,7) = [361.551604; 352.167036; 431.252591; 383.338589; 524.313838; 406.836298;
561.393977; 397.357726; 208.972937];
f(:,7) = [1.636029; 1.665286; 1.576763; 1.590225; 1.578827; 1.577998; 1.502007; 1.512211;
1.610359];
g(:,7) = [2.007543; 2.031260; 2.047239; 2.077505; 2.039920; 1.986055; 2.002346; 2.009576;
2.320092];

%EMG6
x(:,6) = [374; 375; 361; 408; 348; 398; 381; 372; 451];
y(:,6) = [0.947649; 0.591912; 0.703764; 0.472757; 0.685083; 0.417952; 0.838719; 0.529786;
3.886067];
z(:,6) = [322.224623; 295.935802; 310.932284; 236.545633; 353.442106; 246.200752;
332.033997; 257.438672; 3436.600178];
f(:,6) = [1.565215; 1.920311; 1.769655; 2.004073; 1.981547; 1.956077; 1.827496; 1.801226;
1.789529];
g(:,6) = [1.121537; 1.489929; 1.458518; 1.477459; 1.634341; 1.771128; 1.297674; 1.517819;
2.392326];

%EMG5
x(:,5) = [562; 514; 483; 520; 505; 519; 458; 456; 422];
y(:,5) = [0.287129; 0.365564; 0.567300; 0.213592; 0.400782; 0.475114; 0.878513; 0.348591;
0.432334];
z(:,5) = [346.105854; 354.940374; 463.971609; 237.893805; 353.799954; 444.340504;
666.004392; 372.555008; 443.979427];
f(:,5) = [1.634765; 1.673157; 1.837941; 1.689243; 1.841694; 1.853597; 1.747072; 1.650212;
1.850818];
g(:,5) = [2.123758; 1.869824; 1.687827; 1.967657; 1.699850; 1.866835; 1.577558; 2.045154;
2.289426];

%EMG4
x(:,4) = [518; 464; 499; 484; 496; 463; 469; 435; 491];
y(:,4) = [0.992839; 0.724379; 0.750649; 0.868185; 0.950412; 0.676344; 0.469457; 0.545187;
1.224510];
z(:,4) = [1008.351961; 711.824769; 694.860184; 764.881052; 850.295829; 644.881039;
433.081521; 502.845311; 1428.523255];
f(:,4) = [1.616382; 1.601033; 1.611350; 1.640026; 1.522915; 1.561399; 1.679439; 1.640769;
1.731506];
g(:,4) = [1.688667; 1.723751; 1.697187; 1.595170; 1.522617; 1.722084; 1.704875; 1.783236;
2.182968];

%EMG3
x(:,3) = [476; 491; 504; 519; 515; 494; 479; 470; 556];
y(:,3) = [0.864119; 0.815325; 0.803398; 0.607720; 0.824904; 0.791864; 0.869918; 0.909288;
0.724573];
z(:,3) = [816.643237; 735.245239; 737.167179; 555.341705; 758.304779; 831.264590;
880.928591; 811.913063; 829.137985];
```

```
f(:,3) = [1.596277; 1.592738; 1.604615; 1.578898; 1.569518; 1.566613; 1.500097; 1.518783;
1.601866];
g(:,3) = [1.861586; 1.718778; 1.753813; 1.725287; 1.655922; 1.890983; 1.844488; 1.688560;
2.073893];

%EMG2
x(:,2) = [421; 386; 439; 419; 421; 413; 409; 422; 459];
y(:,2) = [0.489709; 0.496419; 0.475615; 0.445918; 0.422851; 0.467722; 0.454503; 0.512714;
0.394246];
z(:,2) = [349.669656; 323.698728; 327.775973; 316.235740; 322.580727; 332.782428;
354.238006; 345.069006; 413.598508];
f(:,2) = [1.845123; 1.785486; 1.835669; 1.692828; 1.812856; 1.807926; 1.786542; 1.832821;
1.813366];
g(:,2) = [1.604205; 1.565568; 1.576704; 1.556862; 1.675068; 1.637632; 1.701409; 1.513925;
2.196049];

%EMG1
x(:,1) = [535; 533; 495; 541; 505; 482; 460; 513; 564];
y(:,1) = [0.724303; 0.753687; 0.695581; 0.480036; 0.529109; 0.589387; 0.649159;
0.534683; 0.247754];
z(:,1) = [690.362653; 702.248253; 594.622623; 474.795354; 457.155135; 506.758411;
485.230268; 469.537923; 284.847717];
f(:,1) = [1.559009; 1.516685; 1.563471; 1.508098; 1.575616; 1.604531; 1.817514; 1.554406;
1.765954];
g(:,1) = [1.788611; 1.736359; 1.685964; 1.791796; 1.730659; 1.733644; 1.702719; 1.719144;
2.100519];

%%Calculate Mahalanobis distances
i = 1;
A = [x(:,1), y(:,1), z(:,1), f(:,1), g(:,1)]; d(i) = mahal(e, A); i = i+1;
A = [x(:,2), y(:,2), z(:,2), f(:,2), g(:,2)]; d(i) = mahal(e, A); i = i+1;
A = [x(:,3), y(:,3), z(:,3), f(:,3), g(:,3)]; d(i) = mahal(e, A); i = i+1;
A = [x(:,4), y(:,4), z(:,4), f(:,4), g(:,4)]; d(i) = mahal(e, A); i = i+1;
A = [x(:,5), y(:,5), z(:,5), f(:,5), g(:,5)]; d(i) = mahal(e, A); i = i+1;
A = [x(:,6), y(:,6), z(:,6), f(:,6), g(:,6)]; d(i) = mahal(e, A); i = i+1;
A = [x(:,7), y(:,7), z(:,7), f(:,7), g(:,7)]; d(i) = mahal(e, A); i = i+1;
A = [x(:,8), y(:,8), z(:,8), f(:,8), g(:,8)]; d(i) = mahal(e, A); i = i+1;
A = [x(:,9), y(:,9), z(:,9), f(:,9), g(:,9)]; d(i) = mahal(e, A); i = i+1;

%%Classify
%Get only distances less than 13.82
di = [];
for i = 1:1:length(d)
    if (d(i) < 13.82)
        di = [di i];
    end
end

[C, I] = min(d);
index = I;


%Use LDA to clarify the solution
% if (length(di) > 2)
%     index1 = di(1);
%     for j = 2:1:length(di)
%         index2 = di(j);
%         A1 = [x(:,index1), y(:,index1), z(:, index1), f(:, index1), g(:, index1)];
```

```
%         A2 = [x(:,index2), y(:,index2), z(:, index2), f(:, index2), g(:, index2)];
%         in = LDA(A1, A2,e);
%         if (in == 2)
%             index1 = index2;
%         end
%     end
%     %sprintf('Most likely EMG%d', index1)
%     index = index1;
% else
%     if (length(di) == 1)
%         %sprintf('Most likely EMG%d', di(1))
%         index = di(1);
%     else
%         %sprintf('Non of the signals')
%         index = 0;
%     end
% end
```

EMG_Error.m

```
clear; clc; close;

%EMG9
x(:,9) = [507; 549; 540; 555; 548; 523; 469; 517; 484];
y(:,9) = [0.218970; 0.203873; 0.206227; 0.156961; 0.201815; 0.186787; 0.275337; 0.207781;
0.166168];
z(:,9) = [194.855554; 212.655886; 210.698086; 177.522100; 200.195273; 215.120384;
216.478495; 191.049127; 193.624315];
f(:,9) = [1.949065; 1.888338; 1.764148; 1.856463; 1.844574; 1.672533; 1.924361; 2.027997;
1.874117];
g(:,9) = [1.816603; 1.951123; 1.934071; 2.025813; 1.911486; 2.078552; 1.681115; 1.889396;
2.323228];

%EMG8
x(:,8) = [543; 604; 615; 612; 607; 619; 561; 615; 874];
y(:,8) = [1.303635; 1.257568; 1.167121; 1.086307; 1.214580; 1.246938; 1.192898; 1.155678;
0.955456];
z(:,8) = [1128.092795; 1205.448441; 1201.459326; 1021.559444; 1163.191022; 1244.015533;
1090.350515; 1141.269490; 1841.943862];
f(:,8) = [1.575190; 1.556878; 1.484764; 1.529495; 1.420569; 1.603564; 1.566915; 1.537945;
1.258479];
g(:,8) = [1.539465; 1.515233; 1.529606; 1.457414; 1.453835; 1.572031; 1.464084; 1.492975;
2.134371];

%EMG7
x(:,7) = [567; 542; 562; 559; 561; 552; 599; 560; 601];
y(:,7) = [0.297111; 0.297484; 0.341431; 0.302723; 0.425680; 0.330622; 0.420241; 0.326378;
0.142905];
z(:,7) = [361.551604; 352.167036; 431.252591; 383.338589; 524.313838; 406.836298;
561.393977; 397.357726; 208.972937];
f(:,7) = [1.636029; 1.665286; 1.576763; 1.590225; 1.578827; 1.577998; 1.502007; 1.512211;
1.610359];
g(:,7) = [2.007543; 2.031260; 2.047239; 2.077505; 2.039920; 1.986055; 2.002346; 2.009576;
2.320092];
```

```
%EMG6
x(:,6) = [374; 375; 361; 408; 348; 398; 381; 372; 451];
y(:,6) = [0.947649; 0.591912; 0.703764; 0.472757; 0.685083; 0.417952; 0.838719; 0.529786;
3.886067];
z(:,6) = [322.224623; 295.935802; 310.932284; 236.545633; 353.442106; 246.200752;
332.033997; 257.438672; 3436.600178];
f(:,6) = [1.565215; 1.920311; 1.769655; 2.004073; 1.981547; 1.956077; 1.827496; 1.801226;
1.789529];
g(:,6) = [1.121537; 1.489929; 1.458518; 1.477459; 1.634341; 1.771128; 1.297674; 1.517819;
2.392326];

%EMG5
x(:,5) = [562; 514; 483; 520; 505; 519; 458; 456; 422];
y(:,5) = [0.287129; 0.365564; 0.567300; 0.213592; 0.400782; 0.475114; 0.878513; 0.348591;
0.432334];
z(:,5) = [346.105854; 354.940374; 463.971609; 237.893805; 353.799954; 444.340504;
666.004392; 372.555008; 443.979427];
f(:,5) = [1.634765; 1.673157; 1.837941; 1.689243; 1.841694; 1.853597; 1.747072; 1.650212;
1.850818];
g(:,5) = [2.123758; 1.869824; 1.687827; 1.967657; 1.699850; 1.866835; 1.577558; 2.045154;
2.289426];

%EMG4
x(:,4) = [518; 464; 499; 484; 496; 463; 469; 435; 491];
y(:,4) = [0.992839; 0.724379; 0.750649; 0.868185; 0.950412; 0.676344; 0.469457; 0.545187;
1.224510];
z(:,4) = [1008.351961; 711.824769; 694.860184; 764.881052; 850.295829; 644.881039;
433.081521; 502.845311; 1428.523255];
f(:,4) = [1.616382; 1.601033; 1.611350; 1.640026; 1.522915; 1.561399; 1.679439; 1.640769;
1.731506];
g(:,4) = [1.688667; 1.723751; 1.697187; 1.595170; 1.522617; 1.722084; 1.704875; 1.783236;
2.182968];

%EMG3
x(:,3) = [476; 491; 504; 519; 515; 494; 479; 470; 556];
y(:,3) = [0.864119; 0.815325; 0.803398; 0.607720; 0.824904; 0.791864; 0.869918; 0.909288;
0.724573];
z(:,3) = [816.643237; 735.245239; 737.167179; 555.341705; 758.304779; 831.264590;
880.928591; 811.913063; 829.137985];
f(:,3) = [1.596277; 1.592738; 1.604615; 1.578898; 1.569518; 1.566613; 1.500097; 1.518783;
1.601866];
g(:,3) = [1.861586; 1.718778; 1.753813; 1.725287; 1.655922; 1.890983; 1.844488; 1.688560;
2.073893];

%EMG2
x(:,2) = [421; 386; 439; 419; 421; 413; 409; 422; 459];
y(:,2) = [0.489709; 0.496419; 0.475615; 0.445918; 0.422851; 0.467722; 0.454503; 0.512714;
0.394246];
z(:,2) = [349.669656; 323.698728; 327.775973; 316.235740; 322.580727; 332.782428;
354.238006; 345.069006; 413.598508];
f(:,2) = [1.845123; 1.785486; 1.835669; 1.692828; 1.812856; 1.807926; 1.786542; 1.832821;
1.813366];
g(:,2) = [1.604205; 1.565568; 1.576704; 1.556862; 1.675068; 1.637632; 1.701409; 1.513925;
2.196049];

%EMG1
x(:,1) = [535; 533; 495; 541; 505; 482; 460; 513; 564];
y(:,1) = [0.724303; 0.753687; 0.695581; 0.480036; 0.529109; 0.589387; 0.649159;
```

```
0.534683; 0.247754];
z(:,1) = [690.362653; 702.248253; 594.622623; 474.795354; 457.155135; 506.758411;
485.230268; 469.537923; 284.847717];
f(:,1) = [1.559009; 1.516685; 1.563471; 1.508098; 1.575616; 1.604531; 1.817514; 1.554406;
1.765954];
g(:,1) = [1.788611; 1.736359; 1.685964; 1.791796; 1.730659; 1.733644; 1.702719; 1.719144;
2.100519];

% %%Plot distributions
% plot3(x(:,1),y(:,1), z(:, 1), 'o', 'Color','r')
% hold on
% plot3(x(:,2),y(:,2), z(:, 2), 'o', 'Color','g')
% hold on
% plot3(x(:,3),y(:,3), z(:, 3), 'o', 'Color','b');
% hold on
% plot3(x(:,4),y(:,4), z(:, 4), 'x', 'Color','r');
% hold on
% plot3(x(:,5),y(:,5), z(:, 5), 'x', 'Color','g');
% hold on
% plot3(x(:,6),y(:,6), z(:, 6), 'x', 'Color','b');
% hold on
% plot3(x(:,7),y(:,7), z(:, 7), '*', 'Color','r');
% hold on
% plot3(x(:,8),y(:,8), z(:, 8), '*', 'Color','g');
% hold on
% plot3(x(:,9),y(:,9), z(:, 9), '*', 'Color','g');
% hold on

% %%Plot distributions
% plot(x(:,1),y(:,1), 'o', 'Color','r')
% hold on
% plot(x(:,2),y(:,2), 'o', 'Color','g')
% hold on
% plot(x(:,3),y(:,3), 'o', 'Color','b');
% hold on
% plot(x(:,4),y(:,4), 'x', 'Color','r');
% hold on
% plot(x(:,5),y(:,5), 'x', 'Color','g');
% hold on
% plot(x(:,6),y(:,6), 'x', 'Color','b');
% hold on
% plot(x(:,7),y(:,7), '*', 'Color','r');
% hold on
% plot(x(:,8),y(:,8), '*', 'Color','g');
% hold on
% plot(x(:,9),y(:,9), '*', 'Color','g');
% % hold on


%Error estimation
c = 0; %Number of wrongly recognized points
num = 0; %Overall points
index = 0; %currently recognized index

[m, n] = size(x);
for j = 1:n
    for i = 1:m
        e = [x(i,j) y(i,j) z(i,j) f(i,j) g(i,j)];
```

```
        index = EMG_classificator(e);
        if (index ~= j)
            c = c+ 1;
        end
        num = num + 1;
    end
    sprintf('Error rate for EMG%d is %.1f%%', j, (c/num)*100)
    num = 0;
    c = 0;
end
```

LDA.m

```
function i = LDA(A1, A2, e)
        %%Transforms factor matrices ->A1(Nx2) and A2(Nx2) using LDA
        %%Return an index ->i [1 or 2], which shows to which class is more
        %%probable to belong the input value -> e

        %%Calculate within class scatter
        S1 = cov(A1);
        S2 = cov(A2);
        S = S1 + S2;

        %%Calculate between class scatter
        m1 = mean(A1);
        m2 = mean(A2);
        m = mean(A1 + A2);
        M1 = (m1-m)*(m1-m)';
        M2 = (m2-m)*(m2-m)';
        M = M1 + M2;

        %%Determine ratio between scatter matrices
        W = inv(S)*M;

        %%Calculate eigen vectors
        [V, D] = eig(W);

        %%Trasform
        A1 = V*A1';
        A2 = V*A2';
        e =  V*e';

        %%Estimate
        d1 = mahal(e', A1');
        d2 = mahal(e', A2');

        if (d1 < d2)
            i = 1;
        else
            i = 2;
        end
```

LDA2.m

```
function i = LDA2(A1, A2, e)
    %%Transforms factor matrices ->A1(Nx2) and A2(Nx2) using LDA
    %%Return an index ->i [1 or 2], which shows to which class is more
    %%probable to belong the input value -> e

    %%Calculate within class scatter
    S1 = cov(A1);
    S2 = cov(A2);
    S = S1 + S2;

    %%Calculate between class scatter
    m1 = mean(A1);
    m2 = mean(A2);

    %Compute the mean of all points outside the first set
    m2_out = mean(A2);
    %Compute the mean of all points outside the first set
    m1_out = mean(A1);

    %Compute the between class scatter of the first set
    M1 = (m1 - m1_out)*(m1 - m1_out)'; %(m1-m)*(m1-m)';
    %Compute the between class scatter of the second set
    M2 = (m2 - m2_out)*(m2 - m2_out)'; %(m2-m)*(m2-m)';
    %M = M1 + M2;

    %%Determine ratio between scatter matrices
    W1 = inv(S)*M1;
    W2 = inv(S)*M2;

    %%Calculate eigen vectors
    [V1, D1] = eig(W1);
    [V2, D2] = eig(W2);

    %%Trasform
    A1 = V1*A1';
    A2 = V2*A2';
    e1 = V1*e';
    e2 = V2*e';

    %%Estimate
    d1 = mahal(e1', A1');
    d2 = mahal(e2', A2');

    if (d1 < d2)
       i = 1;
    else
       i = 2;
    end
```