データ解析

レポート

2018 年 5 月 7 日

理学部　物理・情報科学科　情報コース
学籍番号　15-2202-021-1

北田　和

E-mail アドレス:v021de@yamaguchi-u.ac.jp

1　筋電データの処理

それぞれのプログラム文は以下のようになった。

1.1　筋電位に 1∼40Hz の通過帯域を持つバンドパスフィルター (3 次バターワース) をかける。ただし、単純にバンドパスフィルターをかけるとデータのピーク位置が実際の時刻より少し遅くなることが多い。そこで、順方向と逆方向の両方から一回ずつフィルター処理をすることで時間的なズレを補正する。この処理は、python ならば scipy の fitfit 関数を使えば自動で行われる。

```
import csv
import numpy as np
from scipy import signal
import matplotlib.pyplot as plt
%matplotlib inline
import pandas as pd
from pandas import Series, DataFrame

f = open('PID255_MID11_FileID002_000000_0010.csv',
'r',encoding="shift-jis")
s = open('data_12_2.txt','w',encoding="shift-jis")
i=0
datam=''
dataReader = csv.reader(f)

t, emg, x, y, z= [], [], [], [], []

for row in dataReader:
    if i<10:
        datam=datam+str(row)
    else:
        s.writelines('%s\n' %("   ".join(row)))
        t += [float(row[0])]
        emg += [float(row[1])]
        x += [float(row[2])]
        y += [float(row[3])]
        z += [float(row[4])]

    i=i+1

print(datam)
f.close()
s.close()

n = i-10
dt =0.001
f = 1000
fn = 1/(2*dt)
td = np.linspace(1, n, n)*dt-dt

fp = 1
fs = 40
kagen = 0.01
jougen = 1.0

Wp = fp/fn
Ws = fs/fn

N, Wn = signal.buttord(Wp, Ws, kagen, jougen)
b1, a1 = signal.butter(N,Wn, "low")
y1 = signal.filtfilt(b1, a1, emg)

plt.figure()
plt.plot(td, emg, "b")
plt.plot(td, y1, "r", linewidth=2 , label="butter")
plt.xlim(0,1)
plt.xlabel("Time [s]")
plt.ylabel("Amplitube");

x=0
ef = open('bandpass11_1.txt','w',encoding="shift-jis")
while x < n:
    ef.write('%f   ' %td[x])
    ef.write('%f\n'  %y1[x])
    x+=1
```

1.2　C 言語で筋電位 $E(t)$ を整流する (絶対値をとる)

```
#include<stdio.h>
#include<math.h>
int main(void){
  FILE *rf, *wf;
  double data;
  int i=0;

  rf = fopen("k1_bandpass.txt","r");
  wf = fopen("abs_k1_data","w");

  while((fscanf(rf,"%lf",&data))!=EOF){
    fprintf(wf,"%f    ",fabs(data));
    if(i%2==1)
      fprintf(wf,"\n");
    i++;
  }

  return 0;
}
```

1.3　C 言語で適当な幅 ΔT の窓を設定して、その中での積分値を求め、筋肉の活動度をみる指標とする。つまり、時刻 t における筋肉の活動度 $(a(t))$ は以下で評価することになる。式 (1) これらの処理を行う理由を生データから考察せよ。

$$a(t) = \frac{1}{\Delta T} \int_{t-\Delta T/2}^{t+\Delta T/2} |E(t)| dt \tag{1}$$

```
 1 #include<stdio.h>
 2 #include<stdlib.h>
 3 #include<math.h>
 4 #define BUF_SIZE 256
 5 #define DT 50
 6 #define SAMPLE 1000
 7 void sekibun(double e[],double a[], int n){
 8   int i;
 9   for(i=0; i<DT; i++){
10     a[0]+=(e[i]/SAMPLE);
11   }
12   for(i=1; i<(n-DT); i++){
13     a[i]=a[i-1]+(e[i+DT]-e[i-1])/SAMPLE;
14   }
15 }
16
17 /*------------------------------------------------*/
18 int main(void){
19   FILE *rf, *wf;
20   double data, *e, *a;
21   int n=0, i;
22   char buf[BUF_SIZE];
23
24   if((rf = fopen("abs12_2.txt","r")) ==NULL)
25     return -1;
26   wf = fopen("katudou12_2.txt","w");
27
28   while((fgets(buf,BUF_SIZE,rf))!=NULL){
29     n++;
30   }
31
32
33   fclose(rf);
34   if((rf = fopen("abs12_2.txt","r")) ==NULL)
35     return -1;
36
37   e =(double*) malloc( sizeof(double)*n);
38   a =(double*) malloc( sizeof(double)*(n-DT));
39
40   if(e == NULL || a == NULL){
41     printf("メモリが確保できませんでした。\n");
42     exit(EXIT_FAILURE);
43   }
44
45   for(i=0; i<(n-DT); i++)
46     a[i]=0;
47
48   i=0;
49   while((fscanf(rf,"%lf",&data))!=EOF){
50     fscanf(rf,"%lf",&e[i]);
51     i++;
52   }
53
54   sekibun(e,a,n);
55
56
57   for(i=0; i<(n-DT); i++)
58     fprintf(wf,"%lf    %lf\n",((double)(i+(DT/2))/1000.0),a[i]);
59
60
61   fclose(rf);
62   free(a);
63   free(e);
64   fclose(wf);
65
66   return 0;
67 }
```

2　運動軌道データの切り取り

2.1　データファイルの名前を joints.csv とするとき、このファイルからデータ部のみを取り出した pos-joints.dat と、それ以外 (ヘッダ部とテール部) のみをとりだしたファイル info-joints.dat を作成するシェルスクリプト getdat.sh を作りなさい。各出力ファイルの仕様は以下の通りとする。

```
1 #!/bin/bash
2 if test -f $1.csv; then
3 #    FNAME="$1"
4     egrep [0-9] $1.csv > pos-$1.dat
5     egrep -v [0-9] $1.csv > info-$1.dat
6 else
7     echo "ファイルを指定してください"
8 fi
```

3　運動軌道データの処理

3.1　以下のようなプログラム extract.c を作りなさい。

```
 1 #include<stdio.h>
 2 #include<stdlib.h>
 3 #include<string.h>
 4 #include<math.h>
 5 #define N 20
 6 #define M 256
 7 int main(int argc,char *argv[]){
 8   FILE *rf,*wf[N];
 9   double sample, *data;
10   int makar, i=0,n=1, dmen;
11   char fn[N][M]={'\0'};
12
13
14   if( (sample = atof(argv[1]))==0 ||
(makar = atoi(argv[2])) ==0 || (dmen = atoi(argv[3]))==0 ){
15     printf("サンプリング周波数、マーカ数を数値で入力して
ください。\n\a");
16     i=-1;
17   }
18   else if((rf = fopen(argv[4],"r"))==NULL){
19     printf("ファイルが存在しません。\n\a");
20     i=-1;
21   }
22   if(i==-1){
23     printf("extract <サンプリング周波数> <マーカ数>
<次元> <ファイル名>\n で指定してください。\n");
24     return 0;
25   }
26
27
28
29   for(i=0; i<makar; i++){
30     sprintf(fn[i],"%d-%s",(i+1),argv[4]);
31     wf[i]=fopen(fn[i],"w");
32   }
33
34   data = malloc(sizeof(double)*(dmen*makar+2));
35
36   while(fscanf(rf,"%lf ,%lf ",&data[0],&data[1])!=EOF){
37
38     for(i=0; i<(makar*dmen); i++){
39       if( ( fscanf(rf,",%lf ",&data[i+2]))==EOF ){
40   printf("元ファイルに欠損があります。
出来たファイルは使えません。\n");
41   return 0;
42 }
43 }
44
45     for(i=0; i<makar; i++){
46       fprintf(wf[i],"%lf    ",data[0]/sample);
47     }
48
49     for(i=0; i<(dmen*makar); i++)
50       fprintf(wf[i%makar],"%lf    ",data[i+2]);
51     for(i=0; i<makar; i++)
52       fprintf(wf[i],"\n");
53   }
54
55
56   fclose(rf);
57   for(i=0; i<makar; i++)
58     fclose(wf[i]);
59
60   free(data);
61
62   return 0;
63 }
```

3.2　以下のようなシェルスクリプト extract.sh を作りなさい。その仕様は以下の通りとする。

```
1 #!/bin/bash
2 if test -f $1.csv; then
3     ./extract
$(LANG=ja_JP.sjis grep ‘echo コマ数 | nkf -s‘ info-$1.dat  | nkf -w |cut -f 2 -d "," |cut -f 1 -d "/"|sed -e "s/ //g")
$(LANG=ja_JP.sjis grep ‘echo 計測点数 | nkf -s‘ info-$1.dat  | nkf -w |cut -f 2 -d "," |sed -e "s/ //g") pos-$1.dat
4 else
5     echo "ファイルを指定してくだちい"
6 fi
```

※横幅の関係上改行を行っている

3.3　以下のようなシェルスクリプト cut23 をつくりなさい。

```
1 #!/bin/bash
2 if test -f $1; then
3     cut -f 2- -d " " $1 |  sed -e 's/^[ ]*//g' > ${1/pos/xy}
4 else
5     echo "ファイルを指定してください"
6 fi
```

4　生データの処理

4.1　実験概要

被験者 (19 歳、男性、陸上部所属) の上腕二頭筋、上腕三頭筋に筋電センサを取り付け筋電を、肩、肘、手首 (それぞれマーカ点 1,2,3) の関節ごとに反射マーカを取り付けモーションキャプチャによる計測を行った。この際の体の位置関係を図 1 に示す。なお、図中では青色で示している点が反射マーカの位置である。タスクでは腕の屈筋運動をしてもらった。この際手は拳を握り、ゆっくり屈筋運動した場合とできるだけ早く屈筋運動する場合の二種類を運動開始約 5 秒後から 20 秒間測定した。しかし、以下に示す図では見やすさと、遅い時の手の動きを考慮し、計測開始後 1~4 秒のデータを切り出している。

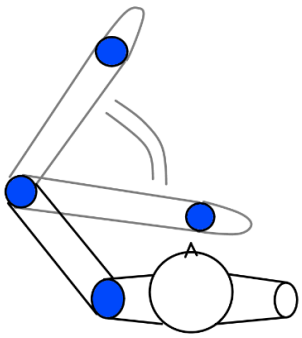

図 1　モーションキャプチャの反射マーカ貼り付け位置イメージ

5　筋電のデータ

5.1　生データ

筋電センサで取得したデータは図 2、3 のようになった。また、以下に示す筋電データの図は速く動かした場合を赤、遅く動かした場合を青で示してあり、横軸が時間、縦軸が EMG である。

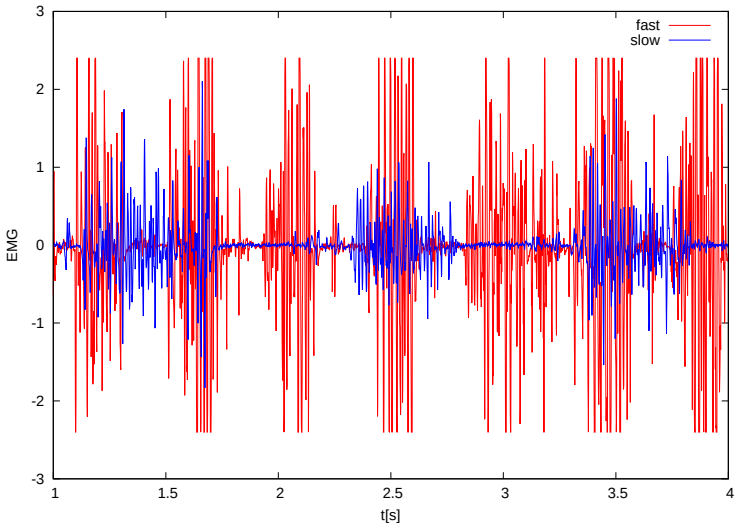

図 2　上腕二頭筋の EMG 生データ

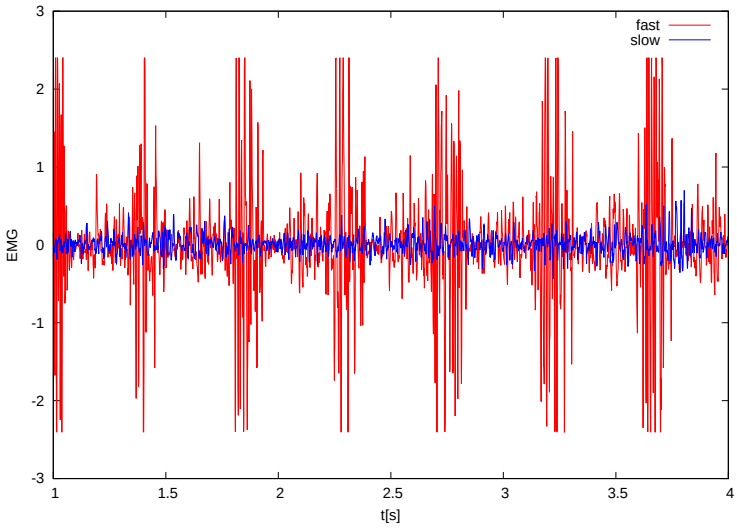

図 3　上腕三頭筋の EMG 生データ

5.2　バンドパスフィルタ後

5.1 のデータに 1.1 の処理を加えた結果を図 4、5 に示す。

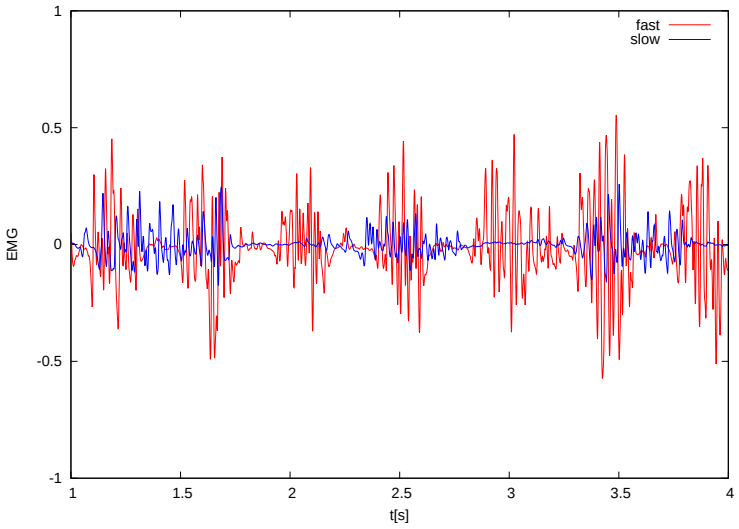

図 4　上腕二頭筋の EMGBPF 後

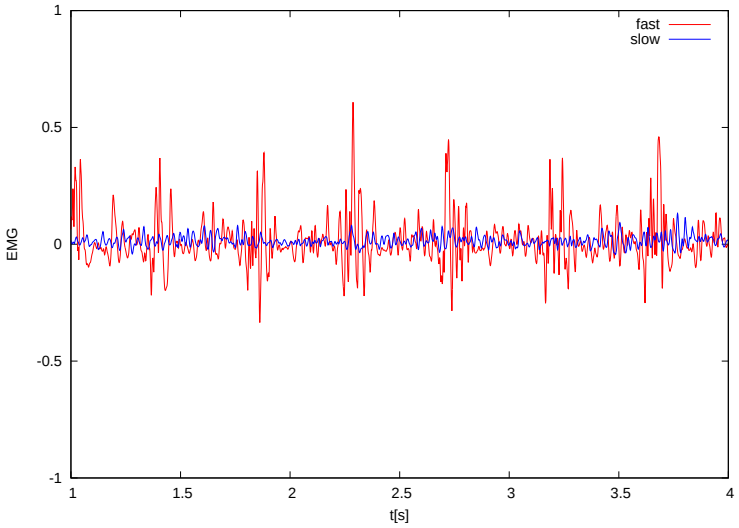

図 5　上腕三頭筋の EMGBPF 後

5.3　整流化後

5.2 のデータを整流した結果を図 6、7 に示す。

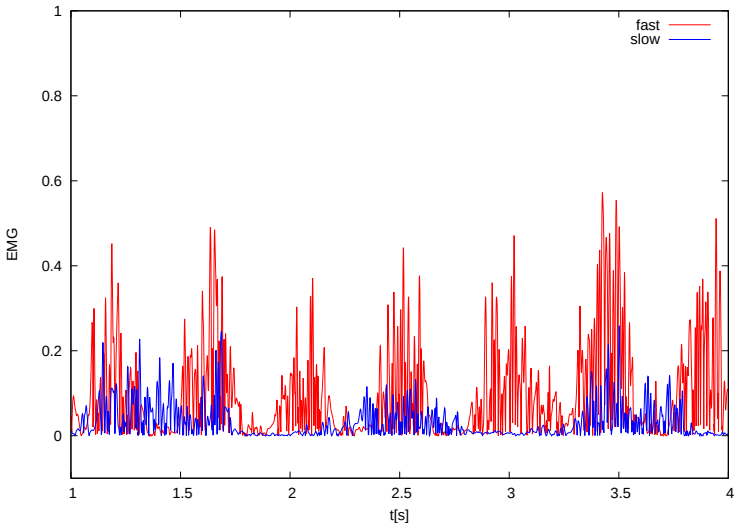

図 6　上腕二頭筋の EMG 整流後

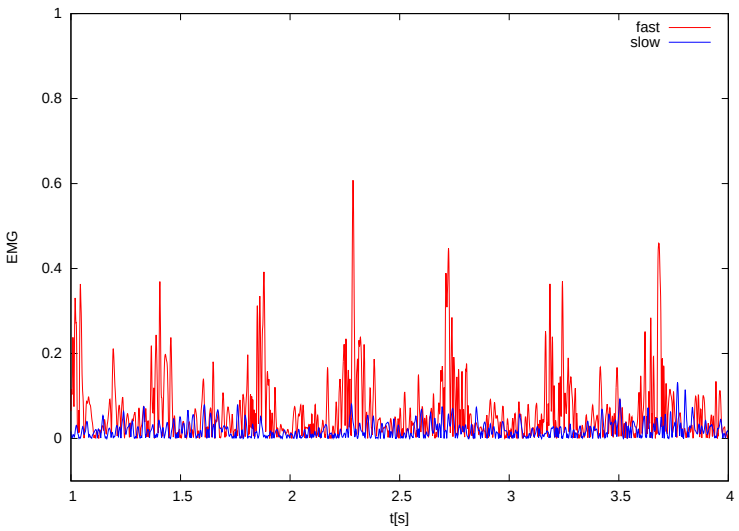

図 7　上腕三頭筋の EMG 整流後

5.4　活動度

5.3 のデータに 1 の処理を $\Delta T = 50$ で加えた結果を図 8、9 に示す。

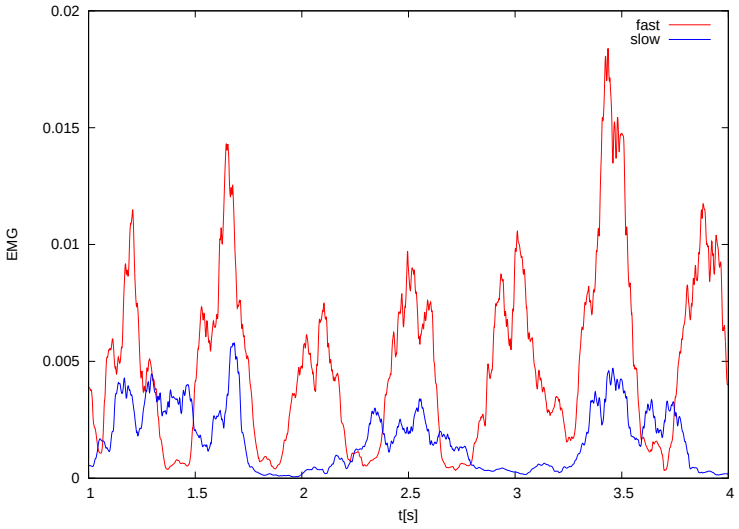

図 8　上腕二頭筋の EMG 活動量

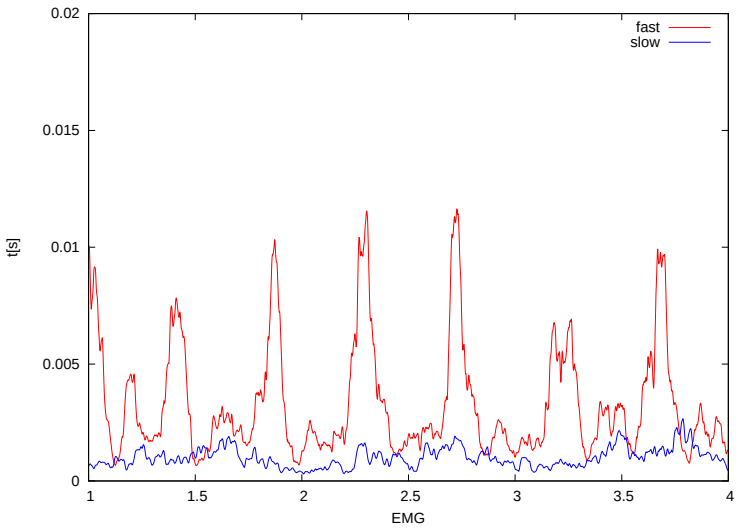

図 9　上腕三頭筋の EMG 活動量

図 2、3 と図 8、9 を比較しても分かる通り、筋肉の活動の時間変化を観察しやすくなっている。例えば、図 8 と図 9 の赤いグラフの比較を行うと双対的に活動量が変化していることがわかる。

6　モーションキャプチャのデータ

6.1　t-x データ

肩の動きの結果を図 10 に、肘の動きを図 11 に、手首の動きを図 12 にそれぞれまとめた。また、以下に示すモーションキャプチャのデータは筋電と同様に速く動かした場合を赤と遅く動かした場合を青で示してある。

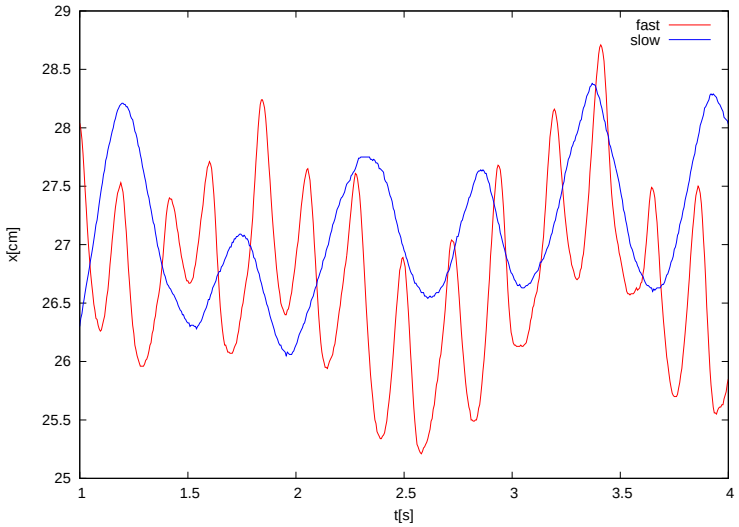

図 10　肩の時間における x 軸方向の動き

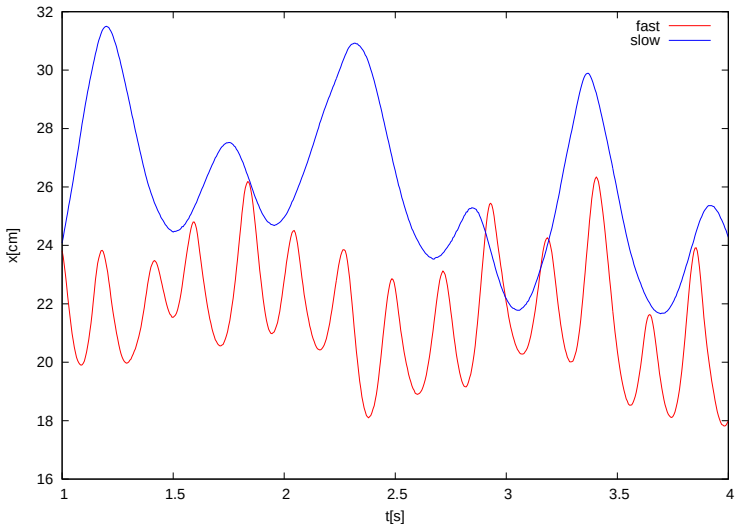

図 11　肘の時間における x 軸方向の動き

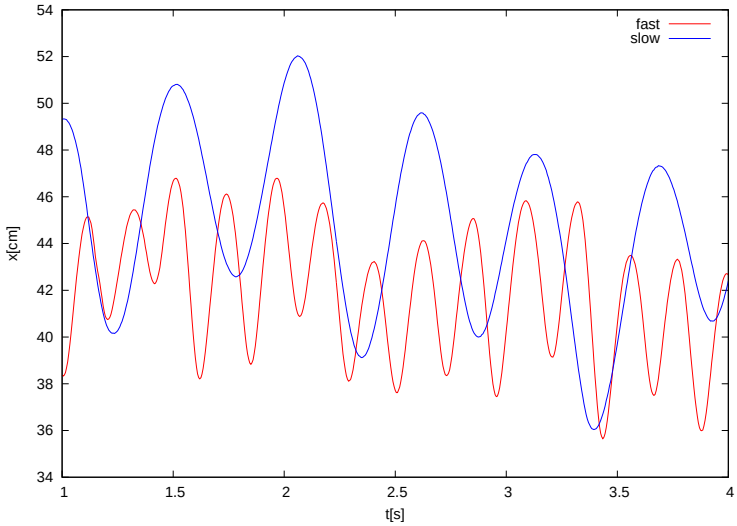

図 12　手首の時間における x 軸方向の動き

6.2　t-y データ

肩の動きの結果を図 13 に、肘の動きを図 14 に、手首の動きを図 15 にそれぞれまとめた。

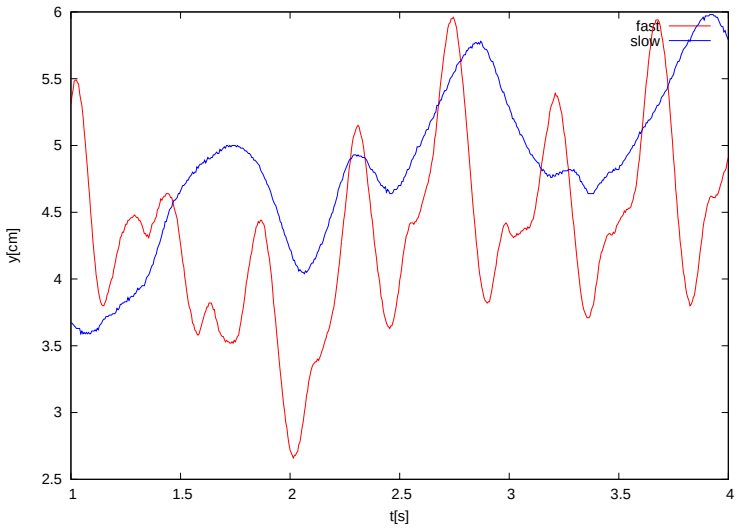

図 13　肩の時間における y 軸方向の動き

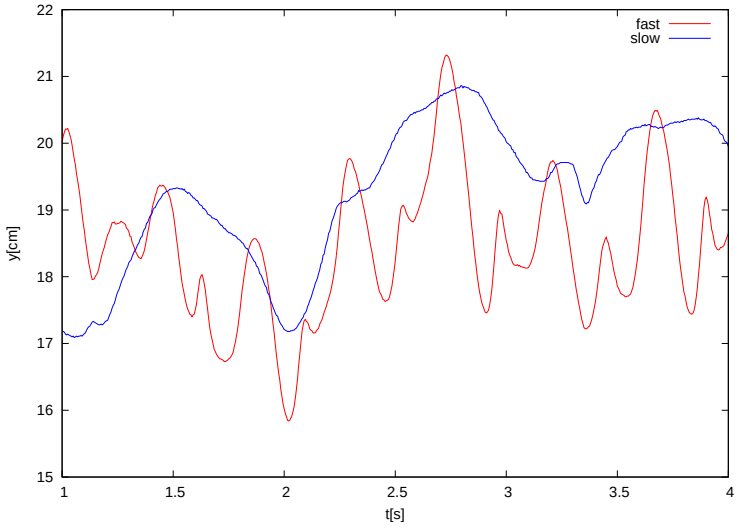

図 14　肘の時間における y 軸方向の動き

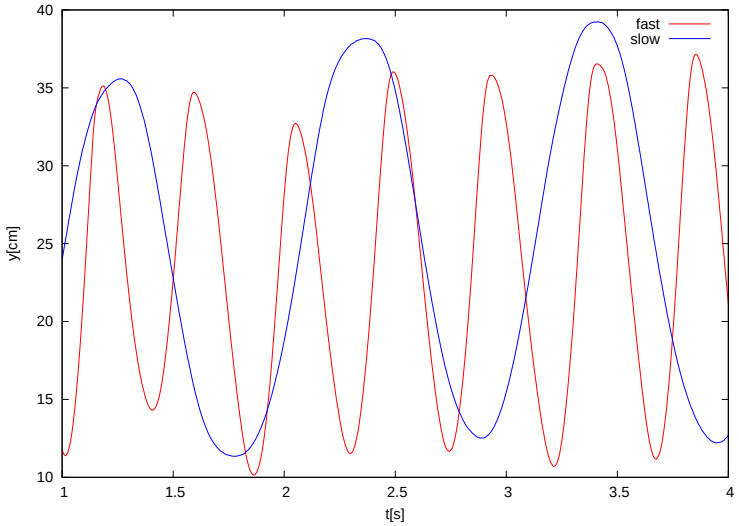

図 15　手首の時間における y 軸方向の動き

6.3　x-y データ

x-y の関係性を図 16 にまとめた。この際の体の位置関係は図 1 と同様である。

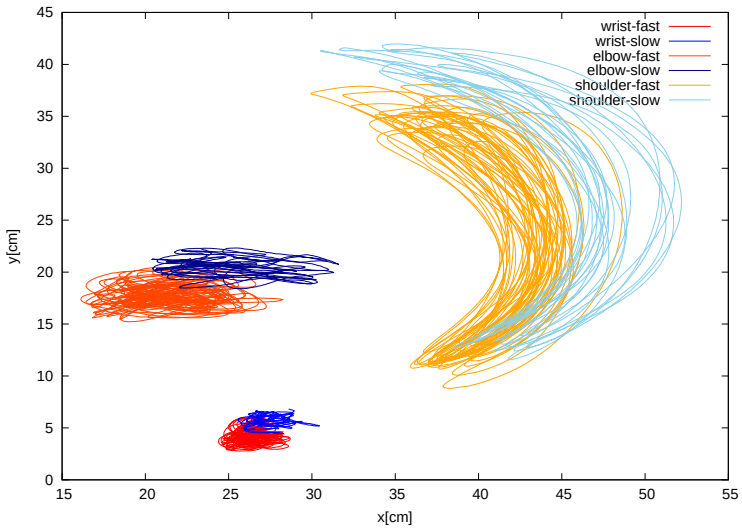

図 16　全体の動きの時間変化

7　考察

図 15 より、1~4 秒の 3 秒間に fast は 7 回、slow は 3 回手首を前後させていることから、それぞれその回数分屈筋運動していることがわかる。この動きに対し、図 10、11、12 では fast は 13~14 回、slow は 5~6 回左右に運動させていることがわかる。よって、1 度屈筋運動をするのに、2 度左右に移動させていることがわかる。また、図 8 の活動量の上がるタイミングと図 15 の前に手が出るタイミングが同じであることから、腕を伸ばすタイミングに上腕二頭筋を使用し、腕を曲げるタイミングに上腕三頭筋を使用していることがわかる。