

【適応フィルタ (LMS アルゴリズム・オンライン) Adaptive Filter LMS Algorithm(Online)】

オンラインでは、 \mathbf{R} , \mathbf{p} が未知であるので、評価関数を $\mathbb{E}[e_k^2]$ から e_k^2 とする。
 $\mathbf{w} = \mathbf{w}_k$ のときの確率勾配は、次のようになる。

$$\nabla[\mathbf{w}_k] = \left. \frac{\partial e_k^2}{\partial \mathbf{w}} \right|_{\mathbf{w}=\mathbf{w}_k} = 2e_k \frac{\partial e_k}{\partial \mathbf{w}_k} = -2e_k \mathbf{x}_k \quad (\text{chain rule})$$

但し、 $e_k = d_k - y_k = d_k - \mathbf{w}_k^T \mathbf{x}_k$ である。
 上式を

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \frac{1}{2} \mu \nabla[\mathbf{w}_k]$$

に代入すると、次のアルゴリズムが成り立つ。

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \mu e_k \mathbf{x}_k$$

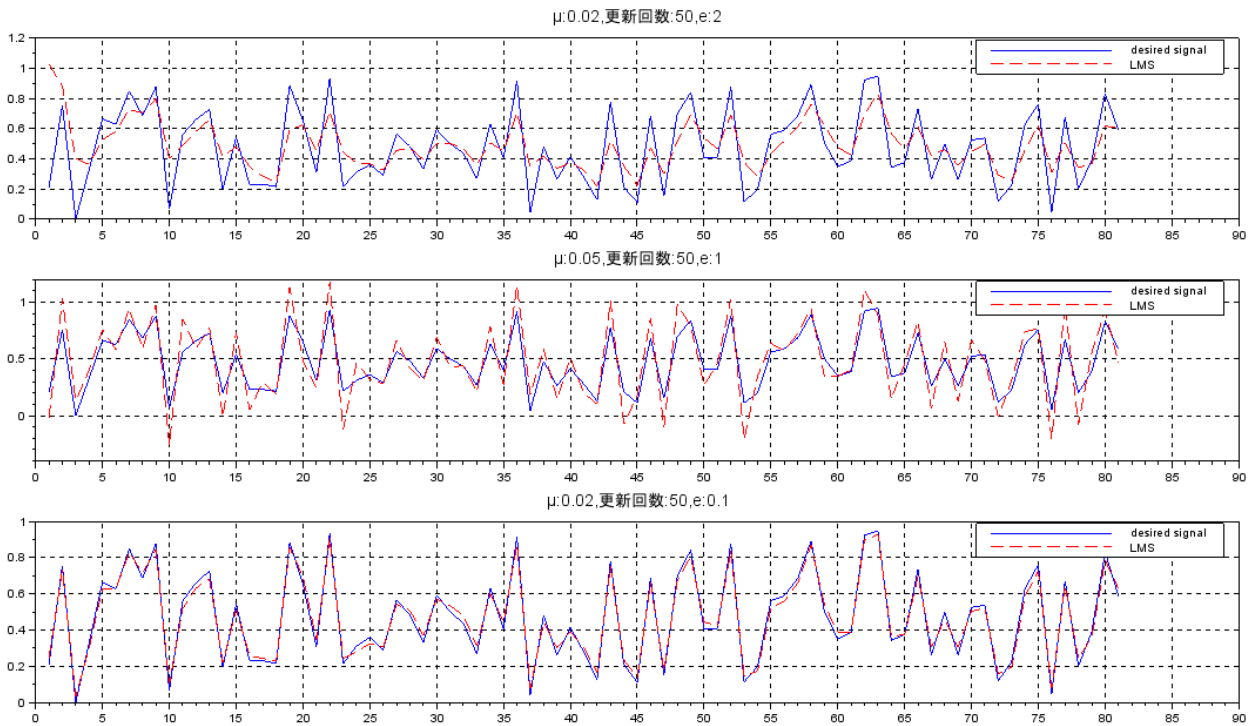


Figure 1: Scilab 実行結果

Source Code 1: Scilab

```

////////////////////////////////////
//  適応フィルタ
//  LMSアルゴリズム(オンライン)
//  Adaptive Filter
//  (LMS Algorithm(Online))
//
//                                     M.Tsutsui
////////////////////////////////////

clear all;

d_size=80;//データサイズ
d=rand(0:1:d_size)';//所望信号
    
```

```

[size_r,size_r2]=size(d);//サイズ数更新

x=rand(size_r,1);//入力信号

w=ones(size_r,1);//適応フィルタ初期係数

funcprot(0);
//-----関数-----
//
// myu:ステップサイズ,update:更新回数
//
// e_con:更新終了条件の数値,y_opt:適応フィルタ出力
//-----

function[y_opt]=lms_opt_cef(myu,update,e_con);

w_buf=[];//係数バッファ
for s_loop=1:1:size_r;//サンプル変化
    for i=1:1:update;//__係数更新ループ__
        y=w'*x;
        e=d(s_loop,1)-y;
        w=w+myu*e*x;
        if (abs(e)<e_con) then,
            w_buf=[w_buf,w];
            break;
        end
    end//-----係数更新ループ__
end

y_opt=w_buf'*x;

endfunction

//_plot-----
subplot(3,1,1)
plot(d);
plot(lms_opt_cef(0.02,50,2),'r--');
xgrid();
legend(['desired_signal','LMS']);
title('μ:0.02,更新回数:50,e:2','fontsize',3);

subplot(3,1,2)
plot(d);
plot(lms_opt_cef(0.05,50,1),'r--');
xgrid();
legend(['desired_signal','LMS']);
title('μ:0.05,更新回数:50,e:1','fontsize',3);

subplot(3,1,3)
plot(d);
plot(lms_opt_cef(0.02,50,0.1),'r--');
xgrid();
legend(['desired_signal','LMS']);
title('μ:0.02,更新回数:50,e:0.1','fontsize',3);

```

Source Code 2: Python

```

#-----
# Module Name: Adaptive Filter
#             LMS Algorithm (online)
# Author m_tsutsui
#-----

```

```

#Library_Import#####
from numpy import*
import math, numpy as np
import matplotlib.pyplot as plt
#Library_Import_end#####

def lms_agm(myu,update,e_con,smp_n):
    """
    myu:step_size,update:update_count

    e_con:end_condition,smp_n:sample_number

    """
    w=np.random.rand(d_size,1) #initial coefficient
    for i in np.arange(1,update+1,1):
        y=matrix(w).T*matrix(x)
        e=d[smp_n,0]-y #error
        w=w+myu*array(e)*x # e-> array(e)
        if(abs(e)<e_con):
            break

    y_opt=matrix(w).T*matrix(x) #ADF out

    return y_opt

if __name__ == '__main__':

    d_size=80
    t=matrix(np.linspace(0,d_size,d_size)).T
    #d=array(sin(t)) #sine wave
    d=np.random.rand(d_size,1)

    x=np.random.rand(d_size,1)

    ADF_out=[]
    for j in np.arange(0,d_size,1):
        ADF_buf=float(lms_agm(0.05,20,0.5,j))
        ADF_out.append(ADF_buf)

#_plot_command_#####
plt.figure(facecolor='w')#Backgroundcolor_white
plt.plot(array(d))
plt.plot(array(ADF_out),"r--")
plt.grid()
plt.legend(('Desired_Signal','LMS'))
plt.title('LMS_Algorithm')
plt.show()
#_end_#####

```