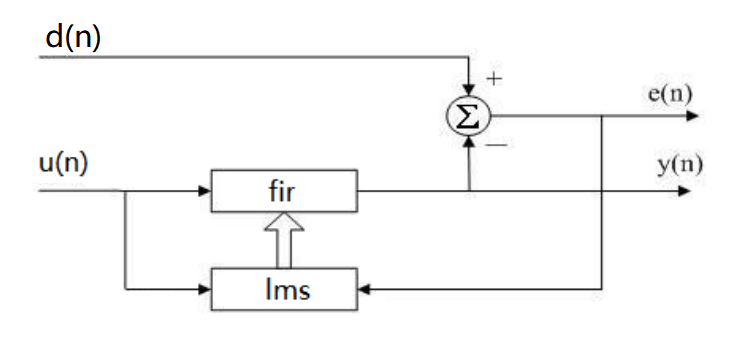
LMS算法下的自适应滤波器



我们使用lms算法对FIR滤波器进行滤波，

本文是为了分析在mui比较小的情况下的LMS算法自适应滤波器的工作下的衰减效果，为了





1. LMS算法单一频率的误差衰减模型

在LMS自适应滤波器的结构上图所示的结构图中，为了分析LMS的衰减速度和输入信号，采样频率、滤波器阶数等等之间的关系，我们将输入信号设置为为简单的单一频率，并且将波形的长度刚刚好是滤波器阶数的整数倍防止频谱泄露，这样简化我们要分析的问题。

下面我们的问题就变成了



本文给出的衰减的模型是



其中amp0是e信号在t=0时候输出的幅度，fs是采样频率，ampn是n信号的幅度，N是滤波器阶数，mui是认为设置的步长，t是时间，f是单一频率信号的频率，phase是信号的相位。

第一项amp0是在t=0的时候e的幅度大小，fir滤波器中的滤波器参数w还没有经过更新的时候，e仍然有输出，此时e的幅度即为amp0，显然，如果w=0，那么amp0=ampd。

第二项为信号衰减项，其中影响这一个项的有fs，ampu，N,mui,也就是说在频率处于指定的范围内，信号的衰减是只和这些东西地方有关的。

第三项是正弦波的信号，其中phase本文不对其进行研究。

1.1单一频率的实验步骤



Matlab demo：

clear;

f\_sample=1e6;

N=100;

mui=1e-7;

time=5;

end\_time=4;

time=[1/f\_sample:1/f\_sample:time];

d=3\*sin(f\_sample/(N/4)\*2\*pi\*time+0);

u=1\*sin(f\_sample/(N/4)\*2\*pi\*time+pi);

% d=3\*ones(size(time));

% u=ones(size(time));

w=zeros(N,1);

% for i = [1:100]

% w(i,1)=i/90;

% end

y=zeros(1,end\_time\*f\_sample);

e=zeros(1,end\_time\*f\_sample);

% w1=zeros(1,end\_time\*f\_sample);

for i=[1:end\_time\*f\_sample]

y(i)=u(i:i+N-1)\*w;

e(i)=d(i+N-1)-y(i);

w=w+mui\*e(i)\*u(i:i+N-1).';

% w1(i)=w(50);

end

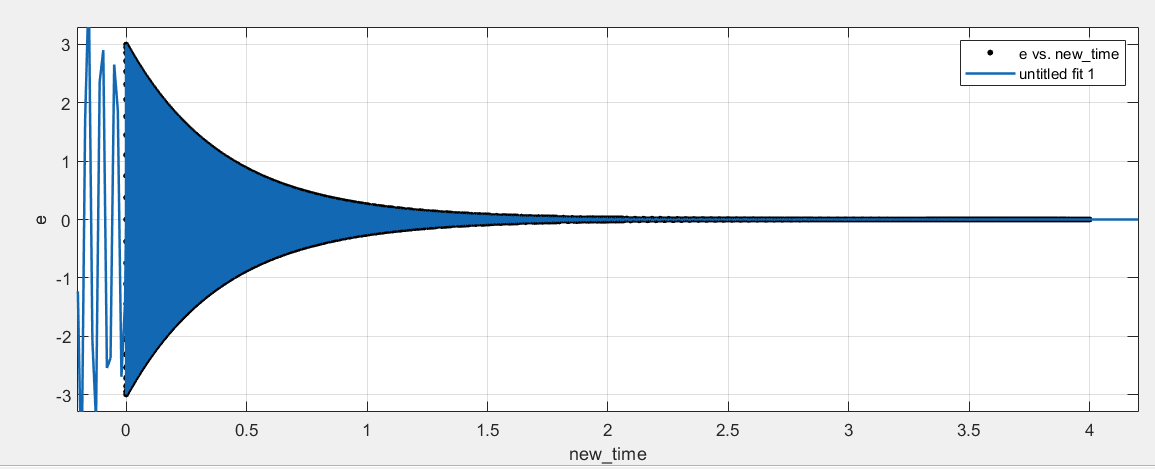
%for cftool

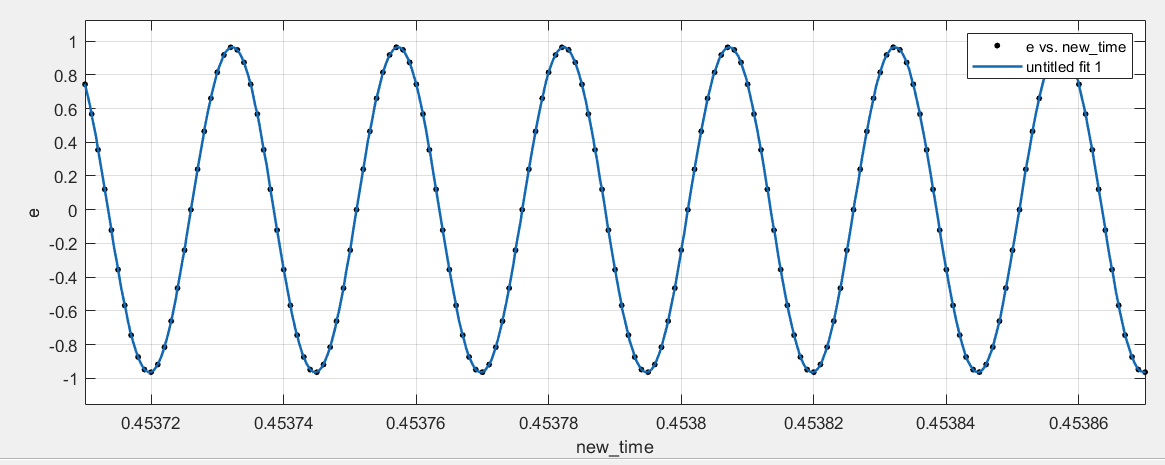
new\_time=time(1:end\_time\*f\_sample);

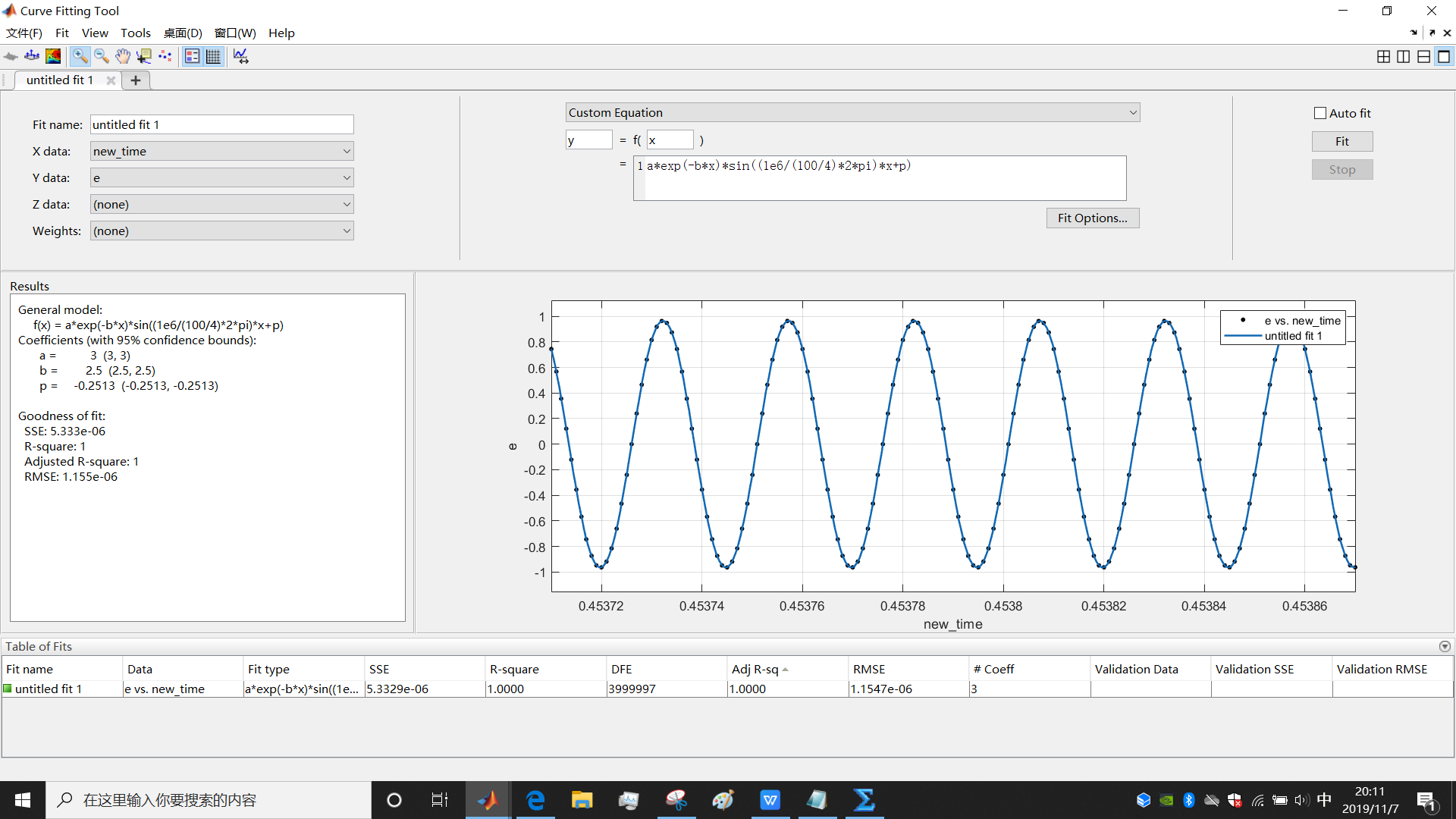
new\_new\_time=new\_time\*f\_sample;

效果：

a\*exp(-b\*x)\*sin(1e6/(100/4)\*2\*pi\*x+p)







1. LMS算法中多个频率的衰减模型



2.1多个频率的实验

Matlab demo

clear;

f\_sample=1e6;

N=400;

mui=3e-8;

time=5;

end\_time=4;

time=[1/f\_sample:1/f\_sample:time];

d=3\*sin(f\_sample/(400/3)\*2\*pi\*time+0)+3\*sin(f\_sample/(400/10)\*2\*pi\*time+0);

u=2\*sin(f\_sample/(400/3)\*2\*pi\*time+pi/3)+1\*sin(f\_sample/(400/10)\*2\*pi\*time+0);

w=zeros(N+1,1);

y=zeros(1,end\_time\*f\_sample);

e=zeros(1,end\_time\*f\_sample);

for i=[1:end\_time\*f\_sample]

y(i)=u(i:i+N)\*w;

e(i)=d(i+N)-y(i);

w=w+mui\*e(i)\*u(i:i+N).';

end

%for cftool

new\_time=time(1:end\_time\*f\_sample);

2.2.建立数学模型

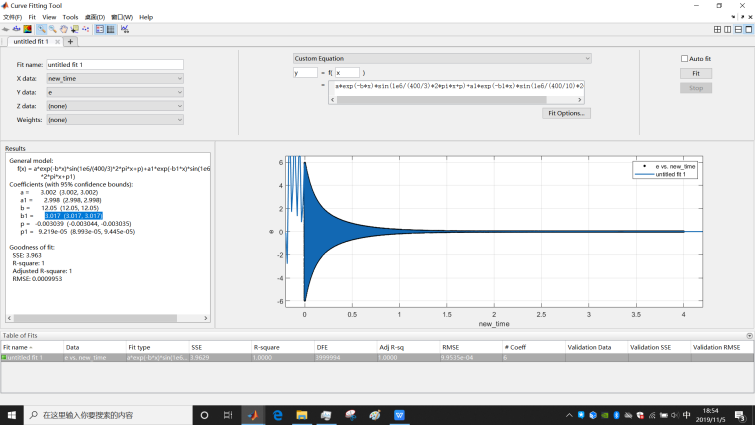


我认为这个解是针对不同频率的解，即在频率上是完全独立的，结论可以推到下面的公式



2.3验证数学模型

a\*exp(-b\*x)\*sin(1e6/(400/3)\*2\*pi\*x+p)+a1\*exp(-b1\*x)\*sin(1e6/(400/10)\*2\*pi\*x+p1)



符合推断。

2.3 应用调节衰减速度

1. lms算法用于e不是收敛于0的情况下的误差模型

自适应算法的实际运用当中，大部分的运用中e最终数值趋向于0，但是也存在一些应用不是是趋向于0 的情况。例如在将lms自适应滤波器运用于噪声滤除当中，e最终输出的是我们需要的信号，在这一种情况下，我们对公式进行修正，至少，我们需要在公式后面加上我们需要的信号，这样才能保证信号最终不是向0收敛。

问题即为如下描述



Matlab demo:

clear;

f\_sample=1e6;

N=400;

mui=3e-8;

time=5;

end\_time=4;

time=[1/f\_sample:1/f\_sample:time];

d=3\*sin(f\_sample/(400/3)\*2\*pi\*time+0)+3\*sin(f\_sample/(400/10)\*2\*pi\*time+0)+3\*sin(f\_sample/(400/7)\*2\*pi\*time+0);

u=2\*sin(f\_sample/(400/3)\*2\*pi\*time+pi/3)+1\*sin(f\_sample/(400/10)\*2\*pi\*time+0);

w=zeros(N+1,1);

y=zeros(1,end\_time\*f\_sample);

e=zeros(1,end\_time\*f\_sample);

for i=[1:end\_time\*f\_sample]

y(i)=u(i:i+N)\*w;

e(i)=d(i+N)-y(i);

w=w+mui\*e(i)\*u(i:i+N).';

end

%for cftool

new\_time=time(1:end\_time\*f\_sample);

* 1. 噪声信号是单一频率信号，需要的信号也是单一频率信号，建立数学模型。

a\*exp(-b\*x)\*sin(1e6/(400/3)\*2\*pi\*x+p)+a1\*exp(-b1\*x)\*sin(1e6/(400/10)\*2\*pi\*x+p1)+a2\*exp(-b2\*x)\*sin(1e6/(400/7)\*2\*pi\*x+p2)

3.2.

3.3.

1. 使用这种办法对进行研究非线性函数（深度学习）
2. 接着修正那个公式



Matlab demo：

clear;

f\_sample=1e6;

N=200;

mui=9e-4;

time=5;

end\_time=0.0005;

time=[1/f\_sample:1/f\_sample:time];

d=3\*sin(f\_sample/(N/20)\*2\*pi\*time+0);

u=1\*sin(f\_sample/(N/20)\*2\*pi\*time+pi);

% d=3\*ones(size(time));

% u=ones(size(time));

w=zeros(N,1);

% for i = [1:100]

% w(i,1)=i/90;

% end

y=zeros(1,end\_time\*f\_sample);

e=zeros(1,end\_time\*f\_sample);

% w1=zeros(1,end\_time\*f\_sample);

for i=[1:end\_time\*f\_sample]

y(i)=u(i:i+N-1)\*w;

e(i)=d(i+N-1)-y(i);

w=w+mui\*e(i)\*u(i:i+N-1).';

% w1(i)=w(50);

end

%for cftool

new\_time=time(1:end\_time\*f\_sample);

Cftool:

a\*exp(-b\*10000\*x)\*sin((1e6/(200/20)\*2\*pi+c\*10000\*exp(-d\*0.01\*x))\*x+p)