数字信号处理课程设计——

基于自适应滤波的语音信号噪声消除

noise elimination of speech signals by technology of adaptive filtering

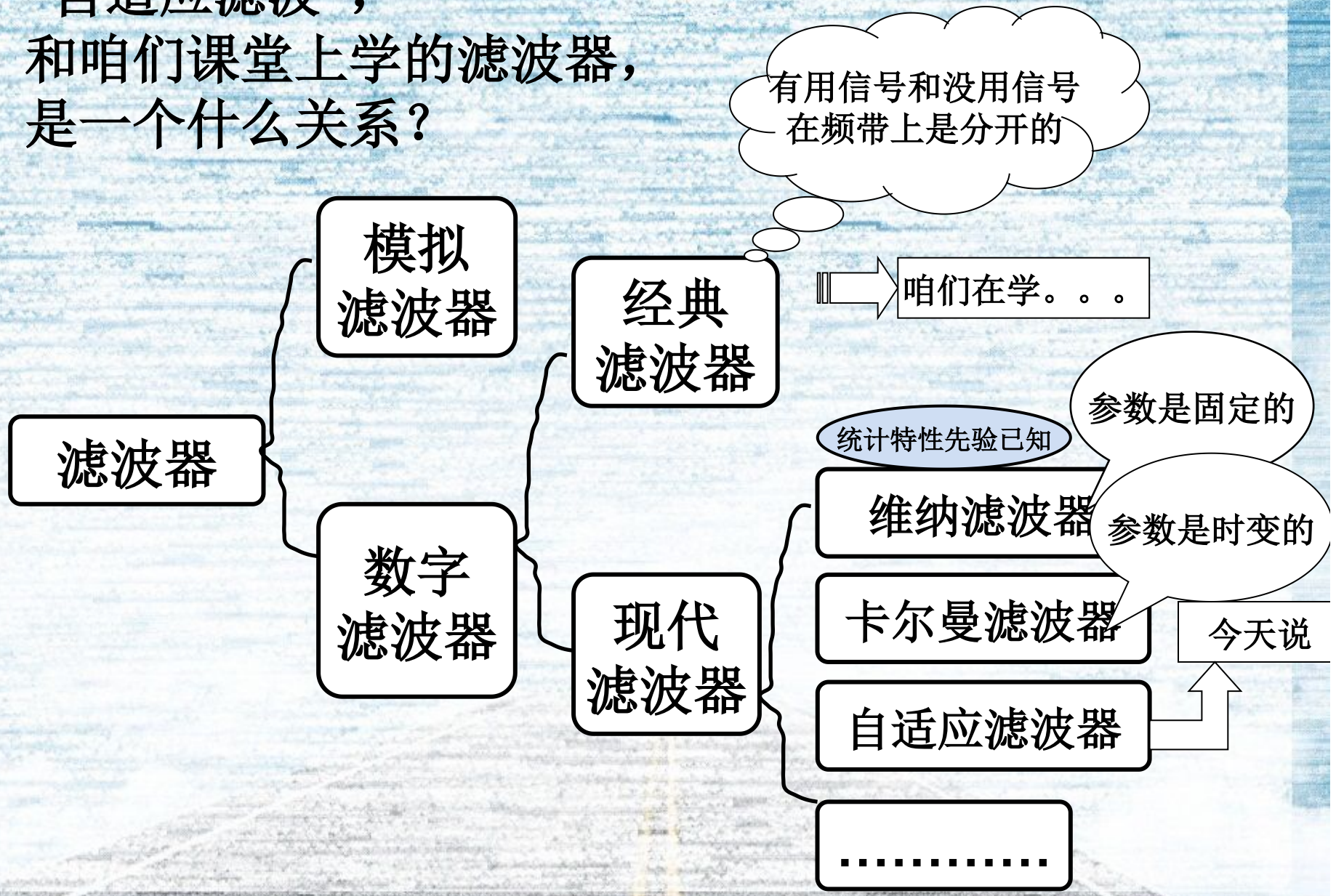
刘北北

2011019060027

目录

- 1、自适应滤波的原理
- 2、关键的算法——原理及实现
- 3、matlab的仿真
- 4、影响滤波效果的因素探究
- 5、自适应滤波的应用之一：去噪
- 6、参考噪声的选取——我的思路
- 7、实际解决问题

“自适应滤波”，
和咱们课堂上学的滤波器，
是一个什么关系？



自适应滤波器的基本结构

基本部件：自适应线性组合器：

设线性组合器的 M 个输入是x(k)，x(k-1)，x(k-2)……x(k-M)，则输出y是这些输入加权后的线性组合。

$$y(k)=\sum_{i=1}^{M}W_i x(k-i)$$

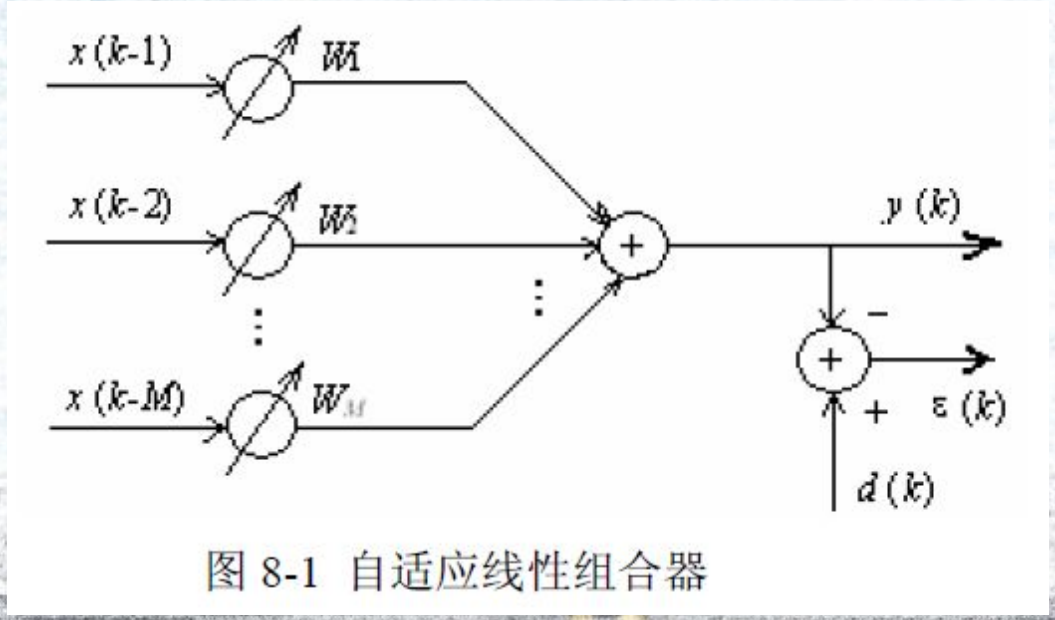
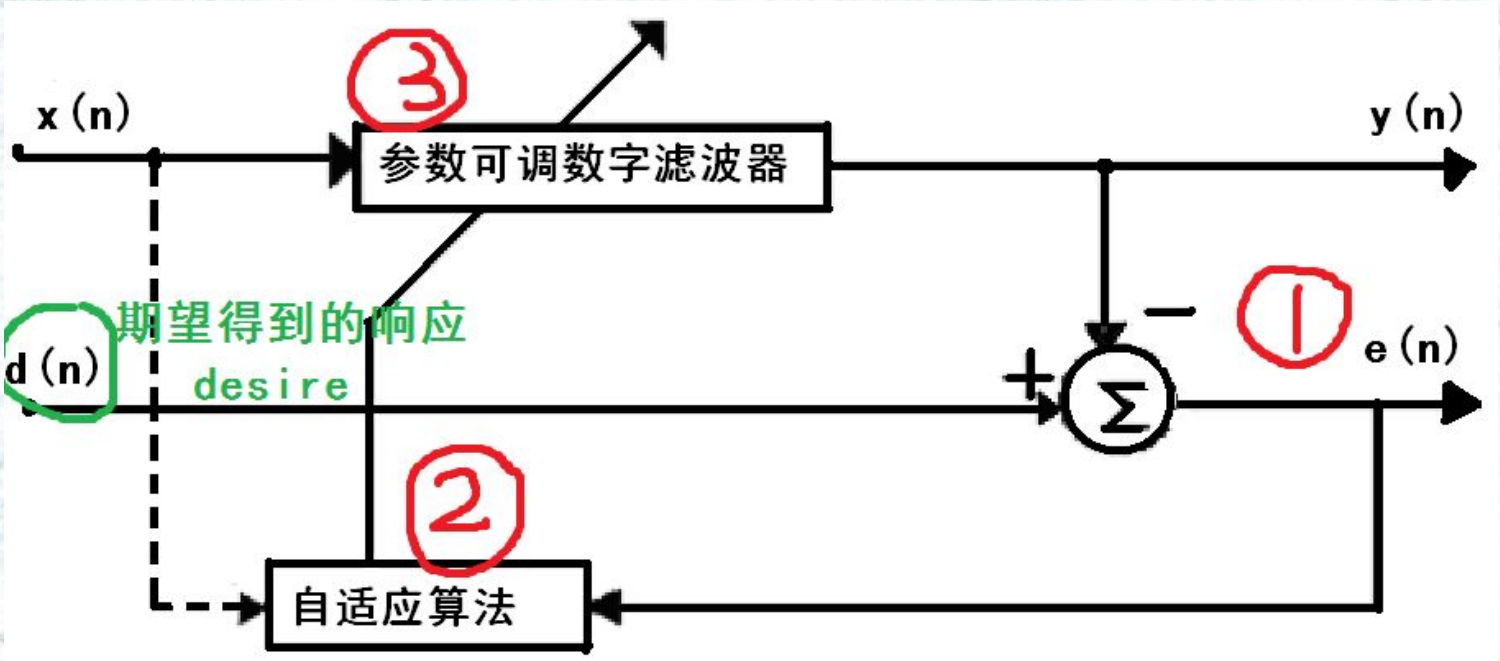


图 8-1 自适应线性组合器

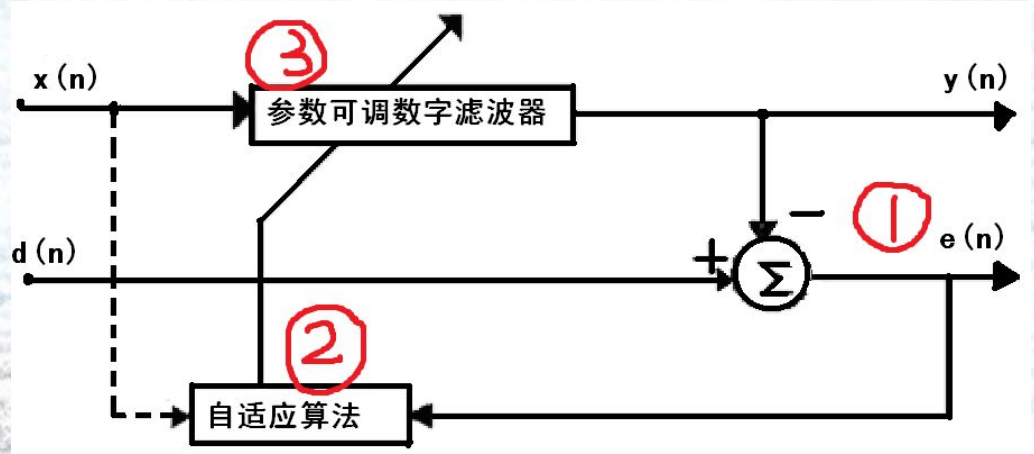
自适应滤波原理

- 1、y(n)与参考信号d(n)进行比较得误差信号e(n)
- 2、通过一种自适应算法和x(n)和e(n)的值来调节参数可调的数字滤波器的参数，即加权系数 ⇨ 具体算法实现
- 3、输入信号x(n)通过参数可调的数字滤波器后得输出信号y(n)

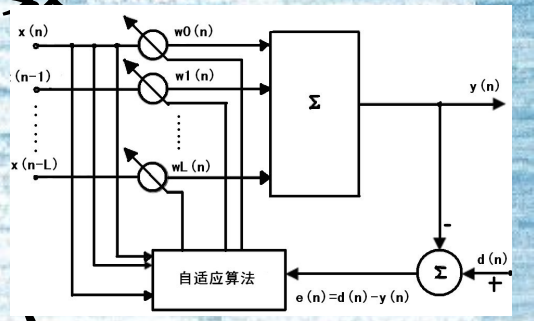


1、y(n)与参考信号d(n)进行比较得误差信号e(n)

- 误差信号：e(n)=d(n)-y(n)
- d(n)为参考信号，y(n)为输出信号。
- e(n)有正有负，平方来评价误差大小
- 误差信号均方值:ε(n)=E[e(n)²]



2、通过一种自适应算法和x(n)和e(n)的值来调节参数可调的数字滤波器的参数，即加权系数



（1）LMS算法（最小均方算法）

误差信号均方值: ʒ(n)=E[e(n)²]

最优化：自变量—>w(n), 因变量—>ʒ(n)

求因变量的最小值：最陡下降法求梯度

$$\nabla(n) \approx \hat{\nabla}(n) = \frac{\partial \hat{\xi}(n)}{\partial \mathbf{w}} = 2e(n)\frac{\partial e(n)}{\partial \mathbf{w}} = -2e(n)\mathbf{x}(n)$$

迭代计算w(n): $\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\nabla(n)$

其中，μ **为控制稳定性和收敛速度的参数**

$$\nabla(n) \approx \hat{\nabla}(n) = \frac{\partial \hat{\xi}(n)}{\partial \mathbf{w}} = 2e(n)\frac{\partial e(n)}{\partial \mathbf{w}} = -2e(n)\mathbf{x}(n)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\nabla(n)$$

由上面两式得： $\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu e(n)\mathbf{x}(n)$

LMS算法的核心：用每次迭代的粗略估计值代替了实际的精确值

（2）RLS算法（递归最小二乘算法）

与之前的LMS算法最主要的区别在于权重W(n)的更新算法不同。

都是“误差最小化”，但评价误差的标准不同。

RLS算法的估计误差准则，是最小二乘时间平均，考虑从零时

刻到当前时刻，n的所有估计误差：

$$e(i) = \mathrm{x}(i) - \hat{\nu}(i)$$

$$\varepsilon(n) = \sum_{i=0}^{n} e(i)^2 = \min$$

为了更好地适应信号变化，加了权值因子λ，叫作“遗忘因子”误差修正为：

$$\varepsilon(n) = \sum_{i=0}^{n} \lambda^{n-i} e^2(i) = e^2(n) + \lambda e^2(n-1) + \cdots + \lambda^n e^2(0)$$

$$\varepsilon(n)=\sum_{i=0}^{n}\lambda^{n-i}e^{2}(i)=e^{2}(n)+\lambda e^{2}(n-1)+\cdots+\lambda^{n}e^{2}(0)$$

引入“遗忘因子”，使得离n时刻较近的赋较大的权重，离n时刻较远的赋较小权重。确保过去的某一段时间内数据被“遗忘”，滤波器工作在平稳状态。

和之前一样，e(n)对W(n)求导，可得最佳的加权系数W(n)

经过数学推导，得递推式：

$$W(n)=W(n-1)+g(n)e(n)$$

- 其中，g(n)表示增益矢量，e(n)表示在n时刻滤波器权重矢量的估计误差

$$g(n)=\frac{P(n-1)u(n)}{\lambda+u^{H}(n)P(n-1)u(n)}$$

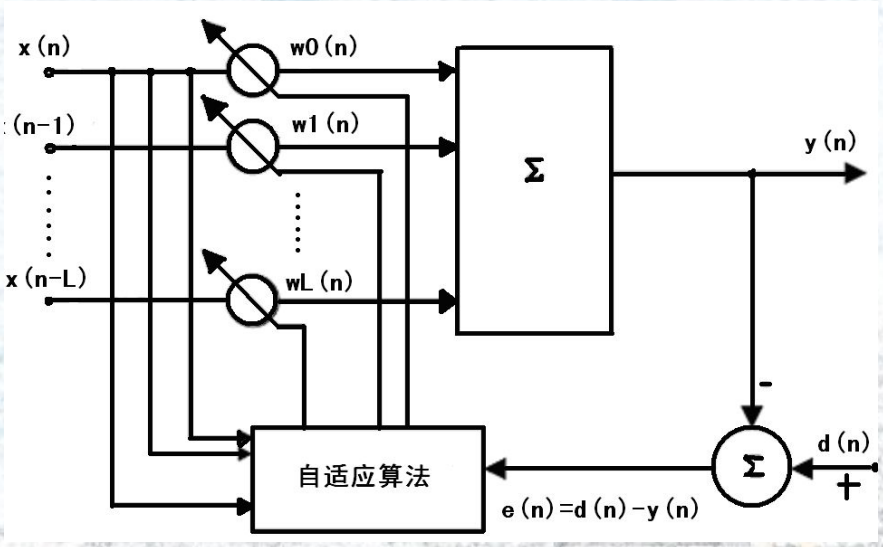
P(n)是自相关矩阵Pxx(n)的逆矩阵

两种算法的比较

- LMS算法（最小均方算法）简便有效，但收敛速度比较慢，不适于快速变化的信号；
- RLS算法（最小二乘算法）收敛快速，稳定，被广泛地应用与实时系统识别和快速启动的信道均衡等领域。但计算量较大，要M^2次运算。

3、输入信号x(n)通过参数可调的数字滤波器后得输出信号y(n)

- 设$x(n)=[x(n)\ x(n-1)\ x(n-2)\ldots\ldots x(n-L)]^T$
- $w(n)=[w_0(n)\ w_1(n)\ w_2(n)\ldots\ldots w_L(n)]^T$
- 其中x(n)为输入信号，w(n)为加权系数
- $y(n)=x(n)^Tw(n)=w(n)^Tx(n)$



算法实现

自适应算法1：LMS算法（最小均方算法）

算法核心： $\mathbf{w}(n+1)=\mathbf{w}(n)+2\mu e(n)\mathbf{x}(n)$

算法步骤：1、初始化W(0)=0

2、更新n=1,2,3，……

3、滤波：y(n)=w(n)Tx(n)

4、误差估计：e(n)=d(n)-y(n)

5、权向量更新：

w(n+1)=w(n)+2ue*(n)x(n)

自适应算法2：RLS算法（最小二乘算法）

算法核心：$W(n)=W(n-1)+g(n)e(n)$

算法步骤：1、初始化$W(0)=0,P(0)=\sigma^{-1}I$ (单位矩阵）

2、更新n=1,2,3，……计算更新增益向量：

$$g(n)=P(n-1)X(n)/[\lambda+X^T(n)P(n-1)X(n)]$$

3、滤波：$y(n)=W(n)^TX(n)$

4、误差估计：$e(n)=d(n)-y(n)$

5、权向量更新：$W(n)=W(n-1)+g(n)e(n)$

更新逆矩阵：

$$P(n)=\lambda^{-1}[P(n-1)-g(n)X^T(n)P(n-1)]$$

P(n)是自相关矩阵Pxx(n)的逆矩阵

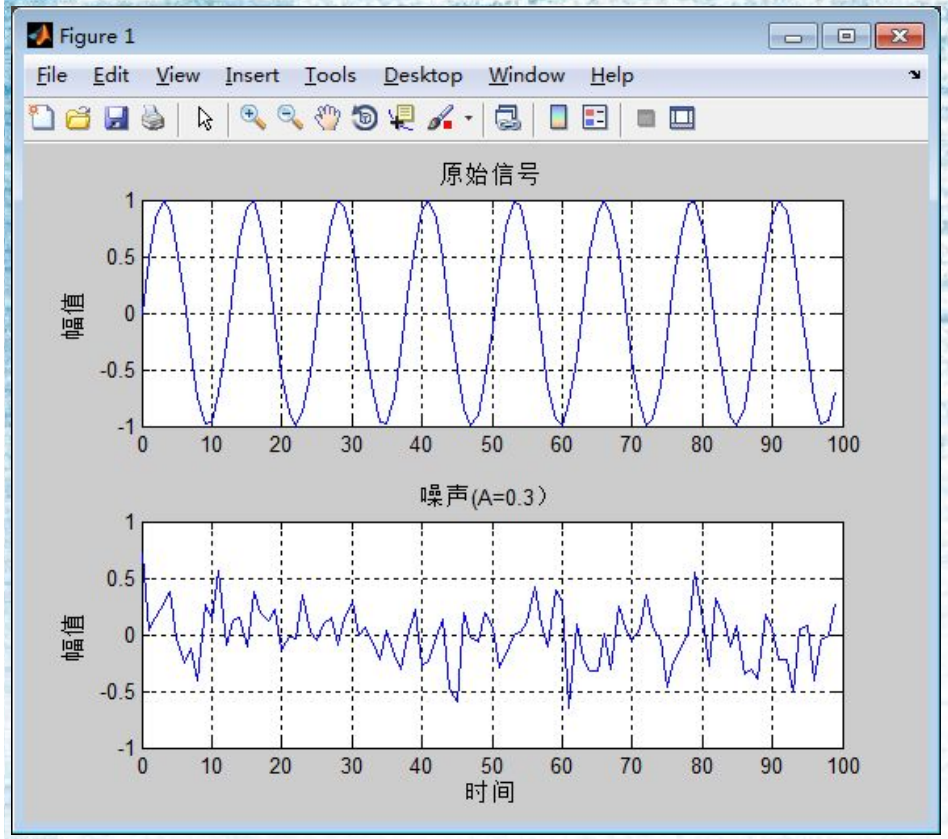
采用LMS（最小均方）自适应算法滤波算法的关键部分代码：

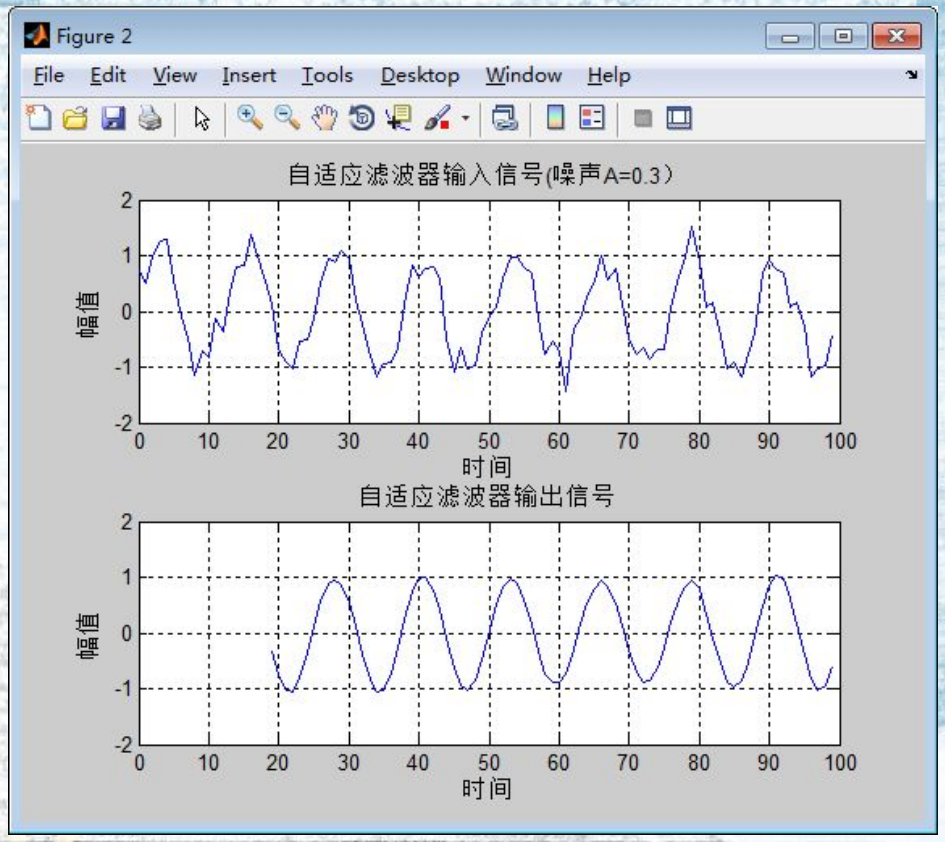
```
% ----------------加了噪声的信号滤波----------------------
xn = xs+xn;
xn = xn.' ;    % 输入信号序列
dn = xs.' ;    % 预期结果序列
M  = 20   ;    % 滤波器的阶数
rho_max = max(eig(xn*xn.'));    % 输入信号相关矩阵的最大特征值
mu = rand()*(1/rho_max)    ;    % 收敛因子 0 < mu < 1/rho
[yn,W,en] = LMS(xn,dn,M,mu);
% 输入参数:
%      xn    输入的信号序列(列向量)
%      dn    所期望的响应序列(列向量)
%      M     滤波器的阶数
%      mu    收敛因子(步长)（要求大于0,小于xn的相关矩阵最大特征值的倒数）
%      itr   迭代次数（默认为xn的长度,M<itr<length(xn)）
% 输出参数:
%      W     滤波器的权值矩阵（大小为 M*itr）
%      en    误差序列(itr*1)
%      yn    实际输出序列(列向量)
```

函数function [yn,W,en] = LMS(xn,dn,M,mu)

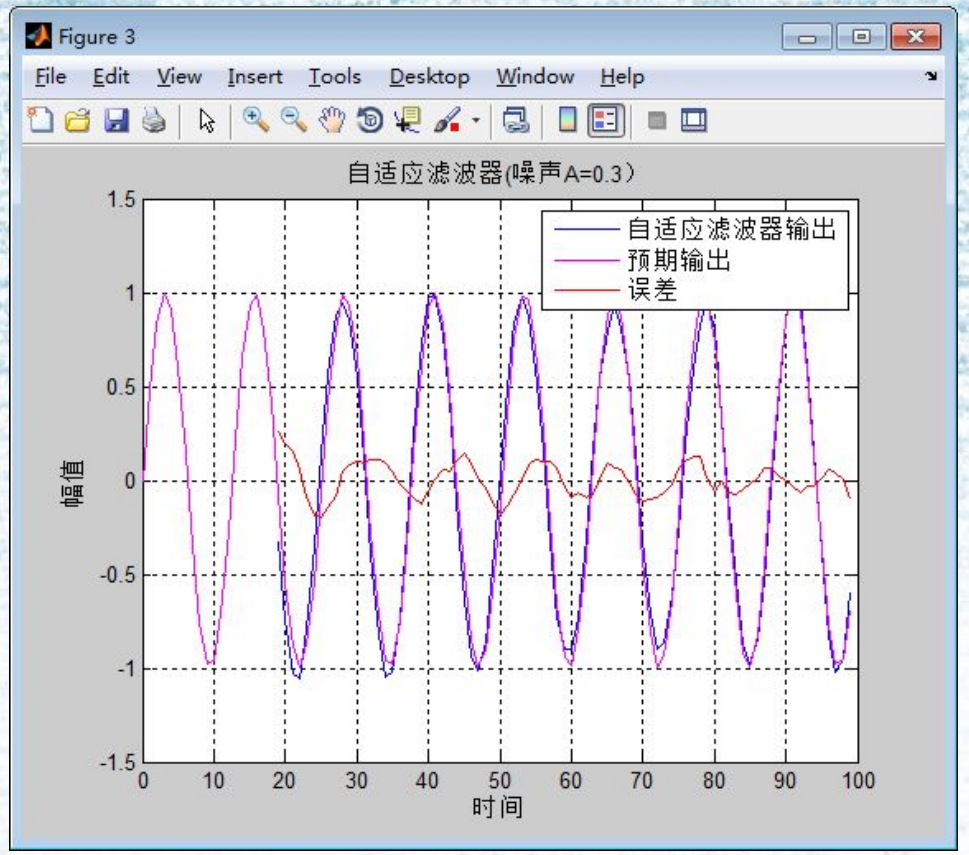
```
%------------ 初始化参数---------------------------------
en = zeros(itr,1);                  % 误差序列,en(k)表示第k次迭代时预期输出与实际输入的误差
W  = zeros(M,itr);                  % 每一行代表一个加权参量,每一列代表一次迭代,初始为0
%------------- 迭代计算---------------------------------
for k = M:itr                       % 第k次迭代
    x = xn(k:-1:k-M+1);             % 滤波器M个抽头的输入
    y = W(:,k-1).' * x;             % 滤波器的输出
    en(k) = dn(k) - y ;             % 第k次迭代的误差
    W(:,k) = W(:,k-1) + 2*mu*en(k)*x; % 滤波器权值计算的迭代式
end
%---------- 求最优时滤波器的输出序列--------------------
yn = inf * ones(size(xn));
for k = M:length(xn)
    x = xn(k:-1:k-M+1);
    yn(k) = W(:,end).'* x;
end
```

原始信号和自适应滤波后的输出





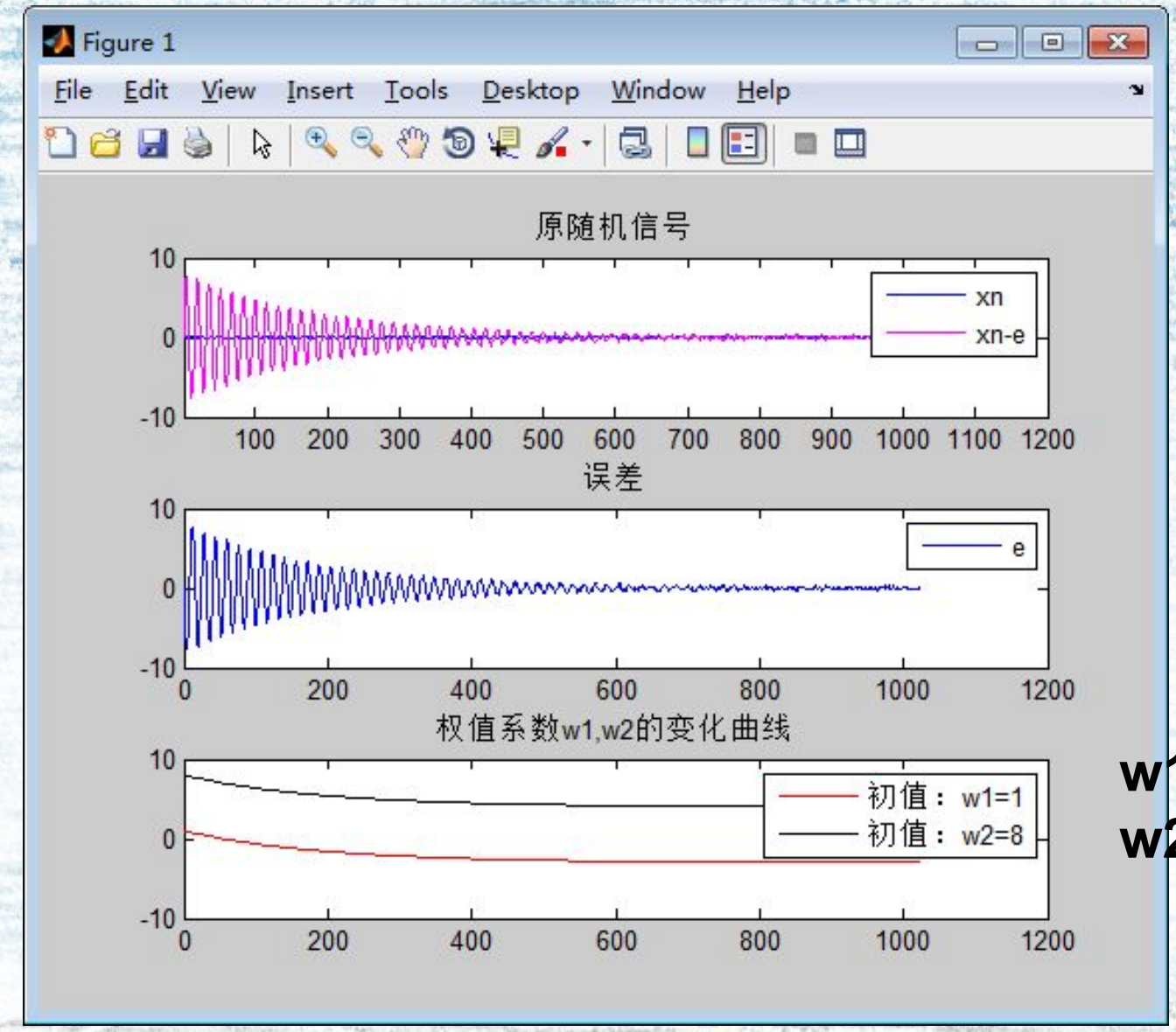
实际输出和预期输出的误差：



哪些因素会影响滤波器的效果？

- w(n)的初值
- u的大小

权值系数W(n)的变化：



w1收敛到-2.9
w2收敛到4.0

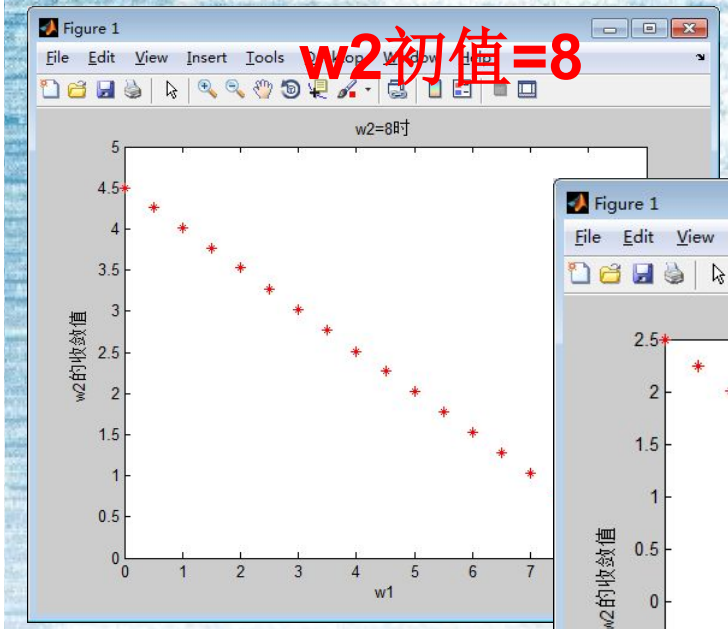
关键部分代码：

```
w1=1*ones(1,1024);
w2=0*ones(1,1024);
u=0.0026;        %u采用网上查的最优系数

for n=2:1024                %1024个数循环移位,每一次循环不考虑n，n只是标号
xn=sin(2*pi*n/16)+0.02*randn(1,1024);    %xn为输入的随机信号
x1(n)=w1(n)*xn(n)+w2(n)*xn(n-1);         %相当于有2个单元的移位寄存器，x1是加权求和的结果
e(n)=xn(n)-x1(n);                        %e(n)是误差，即所得输出x1和期望输出xn之差
w1(n+1)=w1(n)+2*u*e(n)*xn(n);            %更新的迭代式子
w2(n+1)=w2(n)+2*u*e(n)*xn(n-1);
end
plot(1,w2(1,1024),'*r');  %画w2的收敛值，每次画一个，hold on
hold on
```

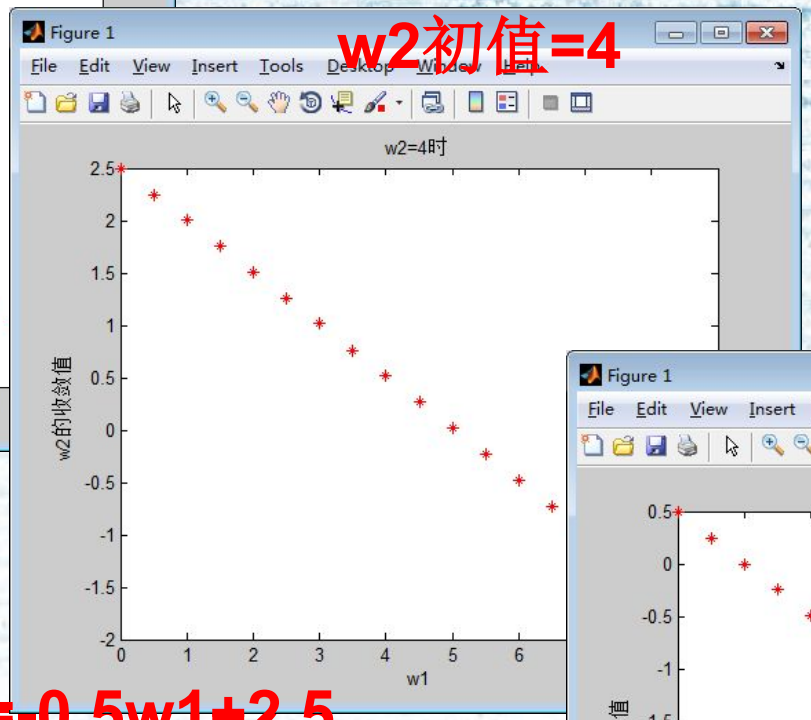
这是一个循环的内部，这部分代码还要循环执行m次。
m=0:9,即w1的值（自变量）

固定W2的初值，看W1初值的变化对W2的收敛值的影响



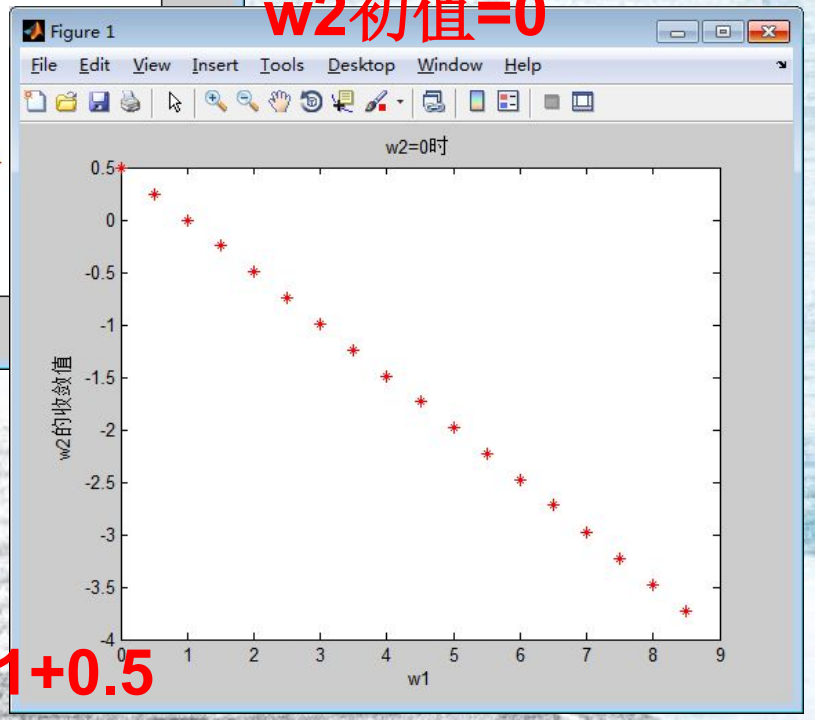
w2初值=8

w2=-0.5w1+4.5



w2初值=4

w2=-0.5w1+2.5

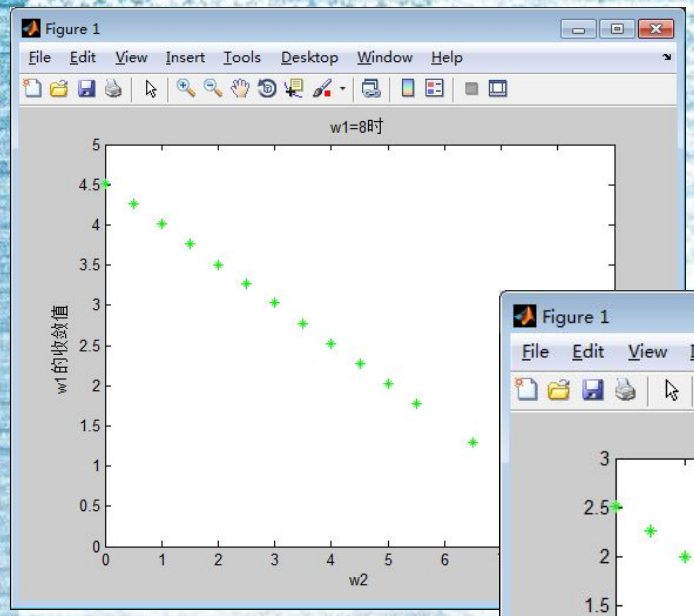


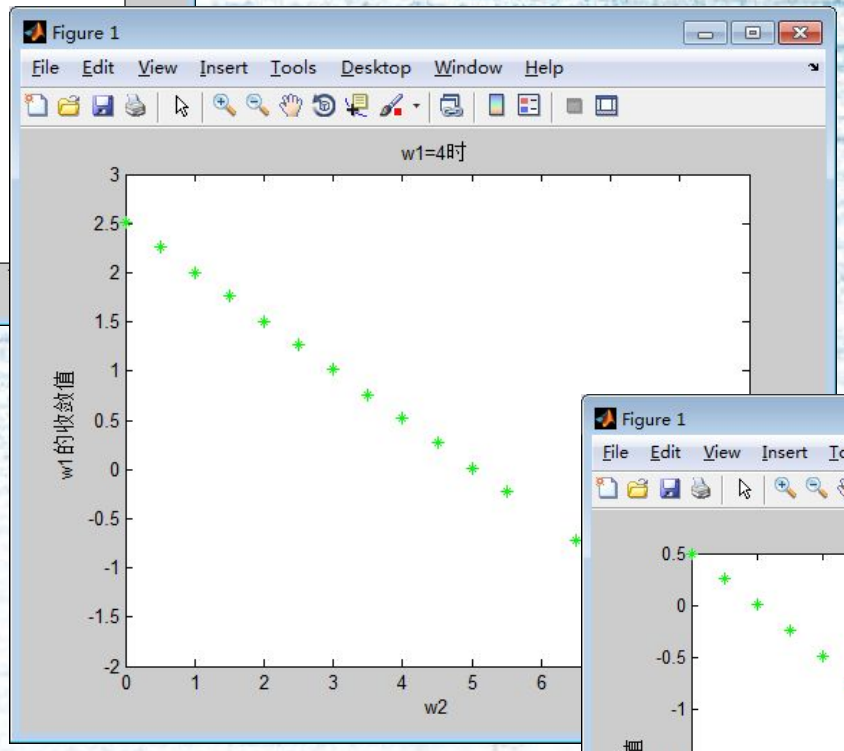
w2初值=0

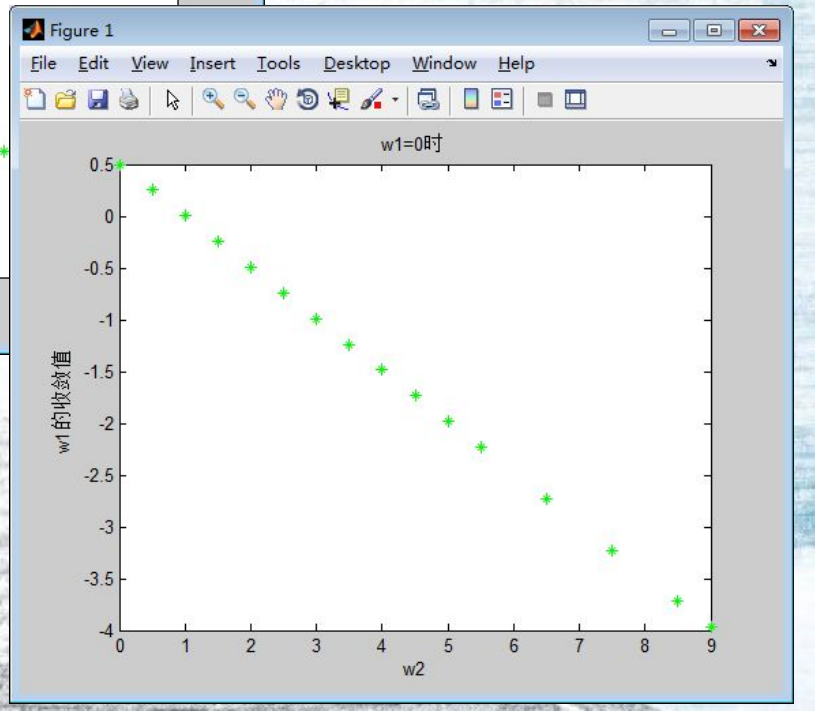
w2=-0.5w1+0.5

$2W_{2收敛}=W_{2初值}--W_{1初值}+1$

两者的关系是对称的







代入公式算后，结果正确

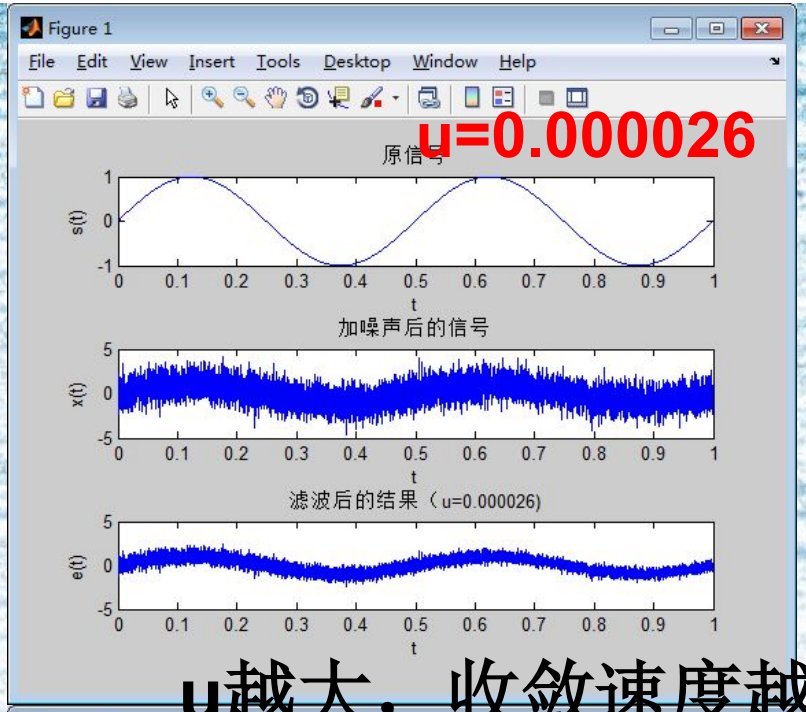
u值的大小：

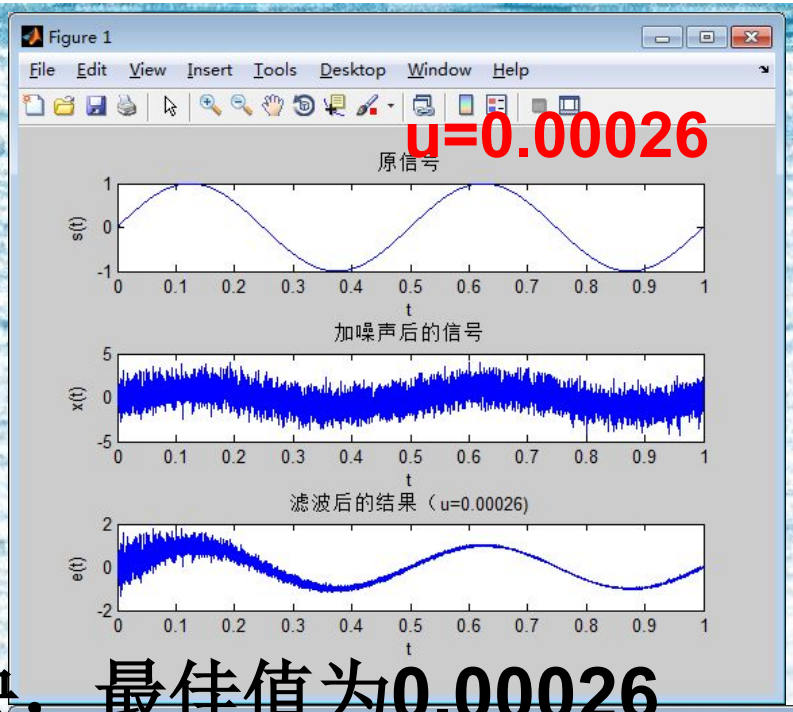
由公式：

$$\mathbf{w}(n+1)=\mathbf{w}(n)-\mu\nabla(n)$$

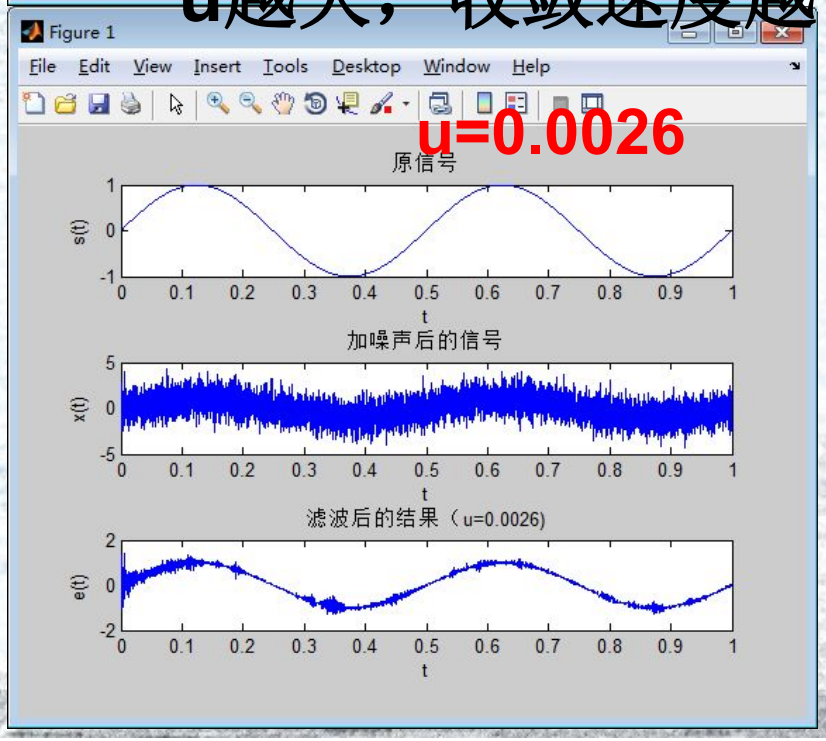
步长参数u的大小，决定每一次迭代中滤波器抽头系数权值的变化大小，

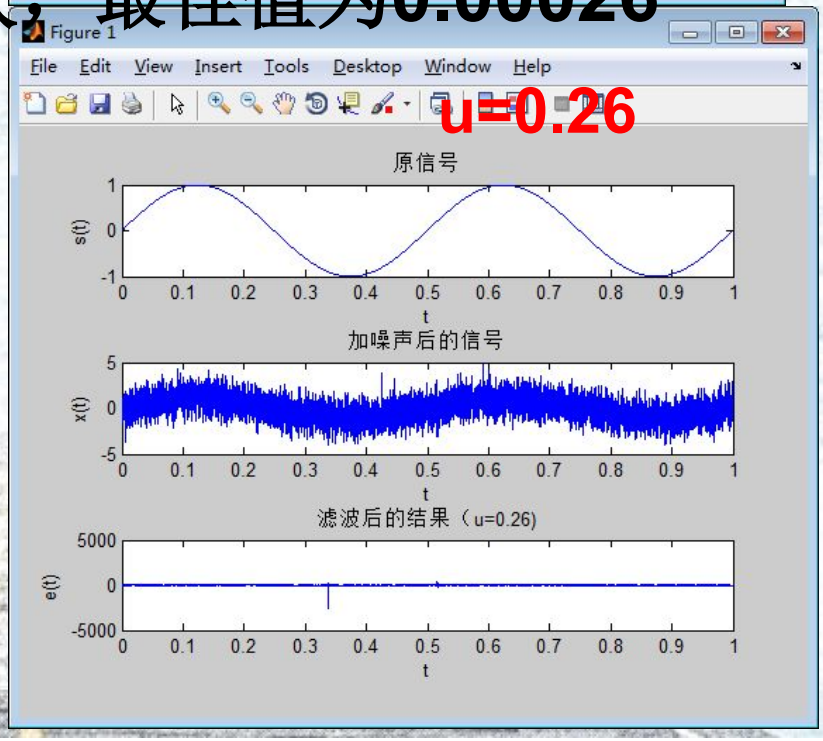
权值向量W(n)的更新，类似于学习的过程，逐渐逼近维纳滤波器的参数。u的大小决定逼近的快慢。





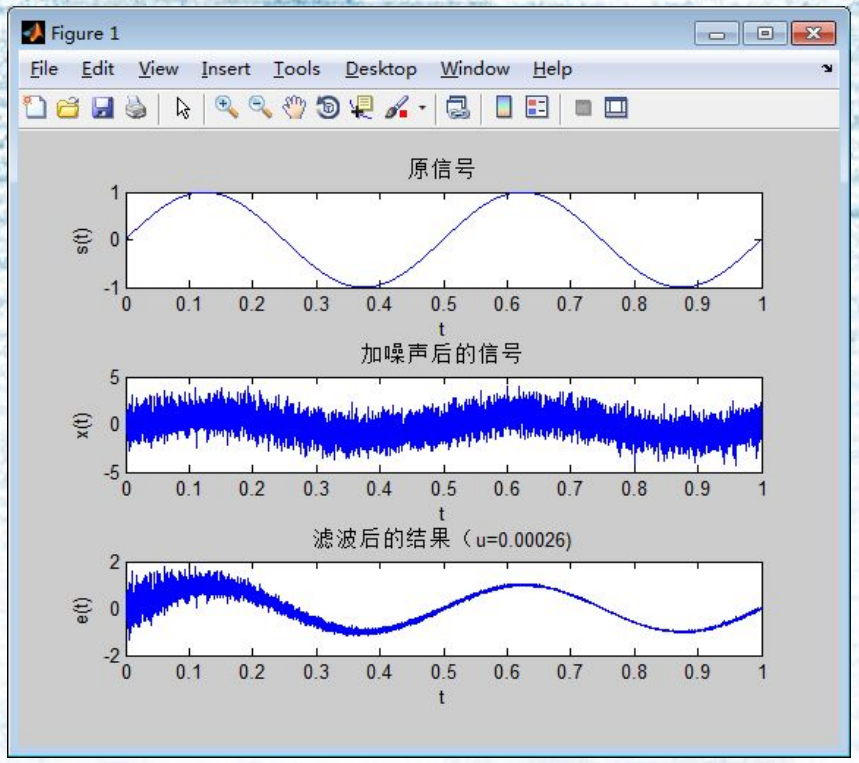
u越大，收敛速度越快，最佳值为0.00026

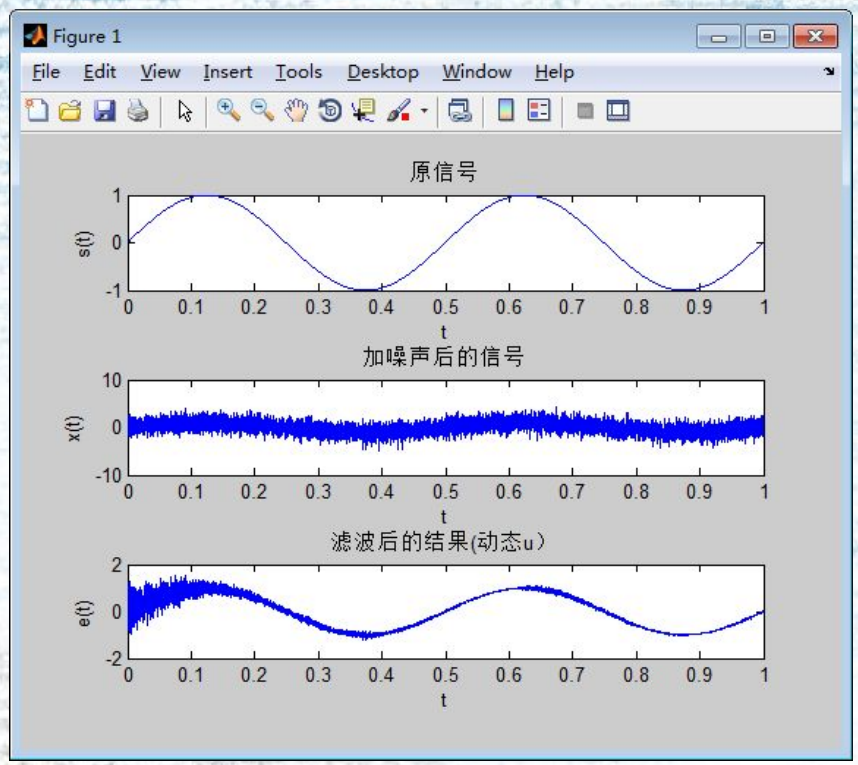




改进办法：
先用大u粗略靠近，再减小u精确逼近

```
if n < 100
    u=0.32;
else
    u=0.15;
end
w = w + x * u * e(n) ;
end
```





进一步自动化：
加速算法的收敛：根据信噪比的情况改变系数

$$w(k+1)=w(k)+2\mu ds[e(k)]x(k)$$

$$ds[e(k)]=\begin{cases}\alpha\ \mathrm{sgn}[e(k)], & |e(k)|>\rho\\ \beta\ \mathrm{sgn}[e(k)], & |e(k)|\leqslant\rho\end{cases}$$

sgn[·]为符号函数，参数α和β是 2 的幂，是用来修正系数向量的

当α>1，增大系数向量的调整；

当α<1，则减小系数向量

β的作用同α，如何选择要视输入信号信噪比情况而定

选择合适的α和β有助于改善算法的收敛特性

图 2　改进的 LMS 算法程序流程图

自适应滤波的应用

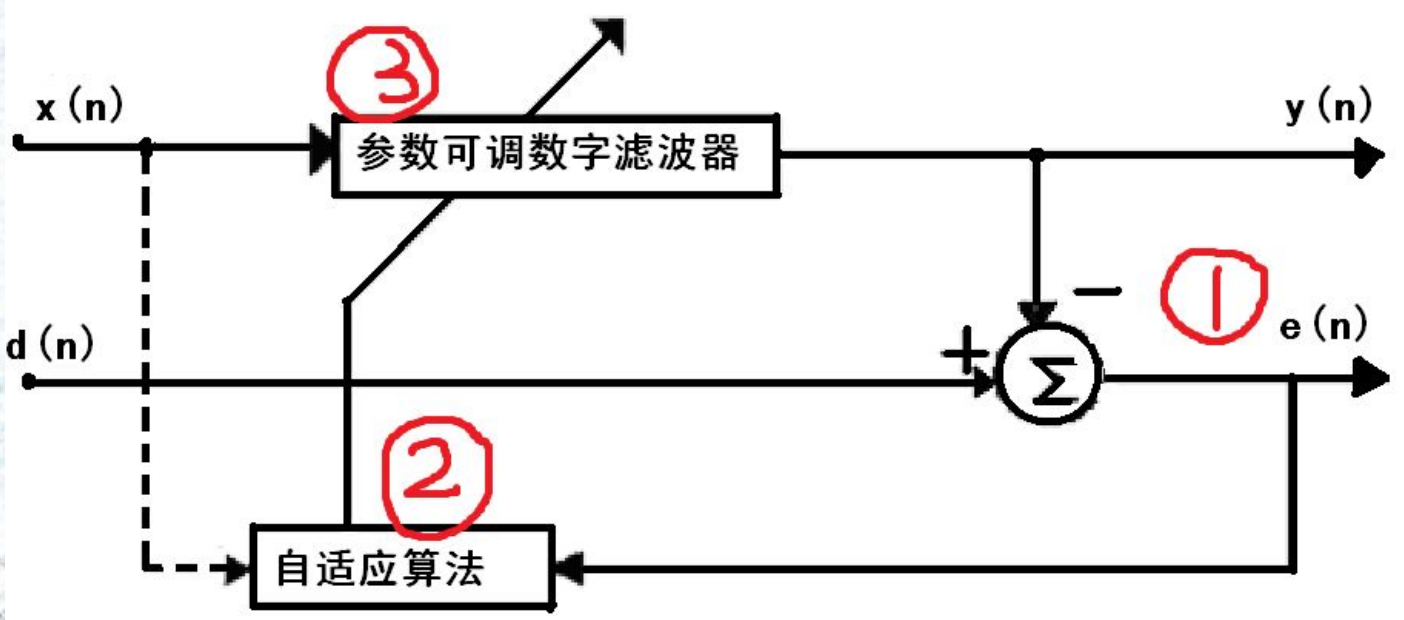
自适应滤波器有哪些应用？

- 1、系统模型识别
- 2、通信信道的自适应均衡
- 3、雷达与声纳的波束形成
- 4、消除心电图中的电源干扰
- 5、噪声中信号的滤波、跟踪、谱线增强以及线性预测等。

自适应噪声消除：

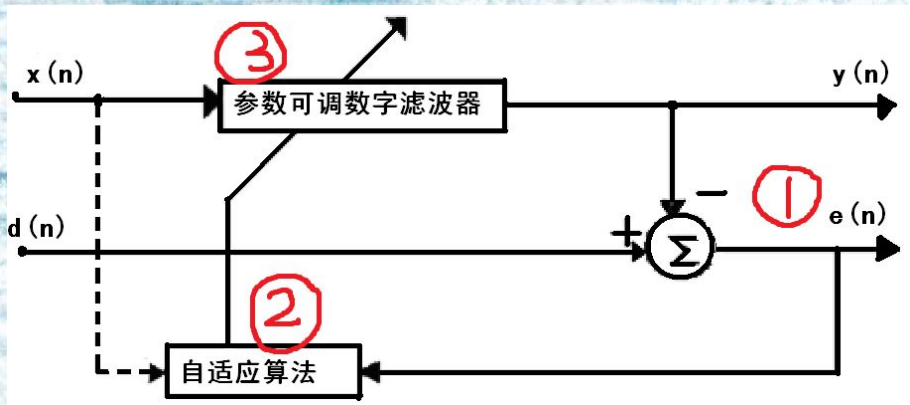
从何而来？？

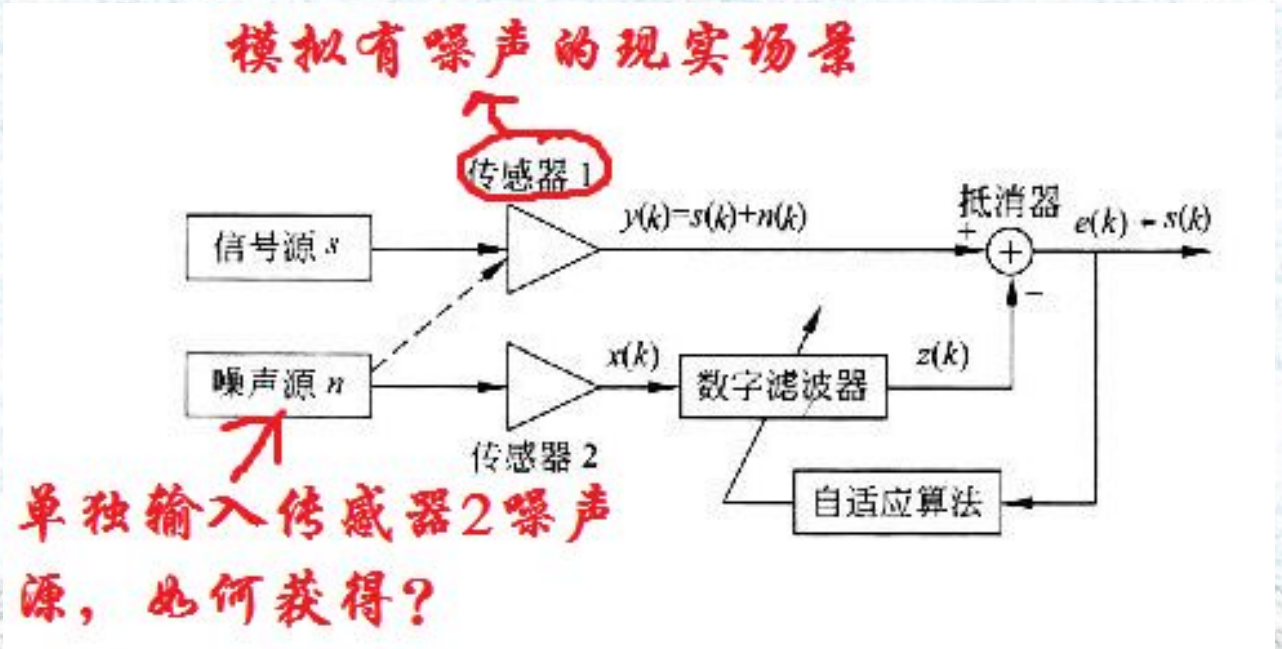
- 1、y(n)与参考信号d(n)进行比较得误差信号e(n)
- 2、通过一种自适应算法和x(n)和e(n)的值来调节参数可调的数字滤波器的参数，即加权系数
- 3、输入信号x(n)通过参数可调的数字滤波器后得输出信号y(n)

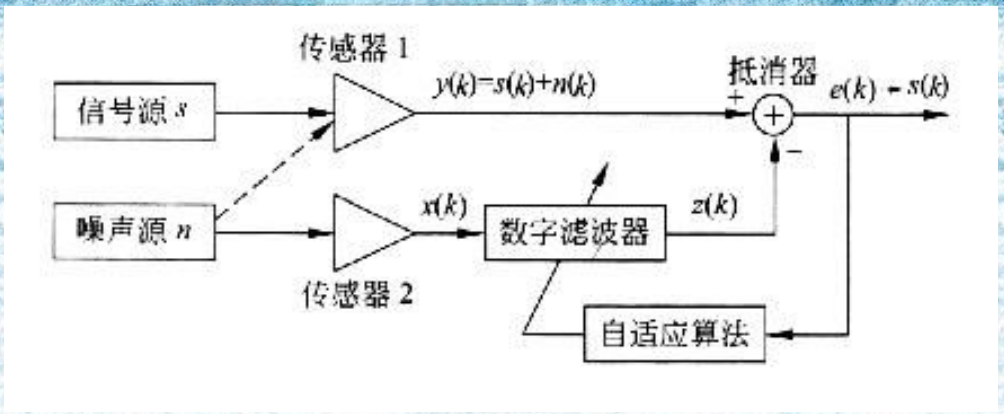


噪声消除的基本思想：

提取噪声，总信号减去噪声







a.主输入端：s+vo：有用信号+干扰噪声

b.参考输入端：vi：与有用信号s无关但和干扰噪声vo相关

利用两噪声信号 vo 和 vi 的相关性，以及参考噪声 vi 和有用信号s 的独立性，使参考噪声 vi 通过自适应滤波器，与主输入中噪声分量逼近并相减，输出误差信号 e 。

自适应滤波算法,决定滤波器对参考信号 vi 的处理,使滤波器的输出尽可能逼近主输入中的干扰成分。

在最佳准则下，滤波器的输出 v 逼近 vo <==>系统输出 e 逼近 s

我的思路：

在新闻中心时，在家园餐厅采访学长学姐，环境嘈杂，录音的背景噪声特别大，影响效果。

希望能用滤波的方法来去掉噪声，又尽量小地破坏人说话的声音。

虽然效果差，但仍可以听出来学姐的声音，说明耳朵具有分辨两种声音的能力，相信计算机也有

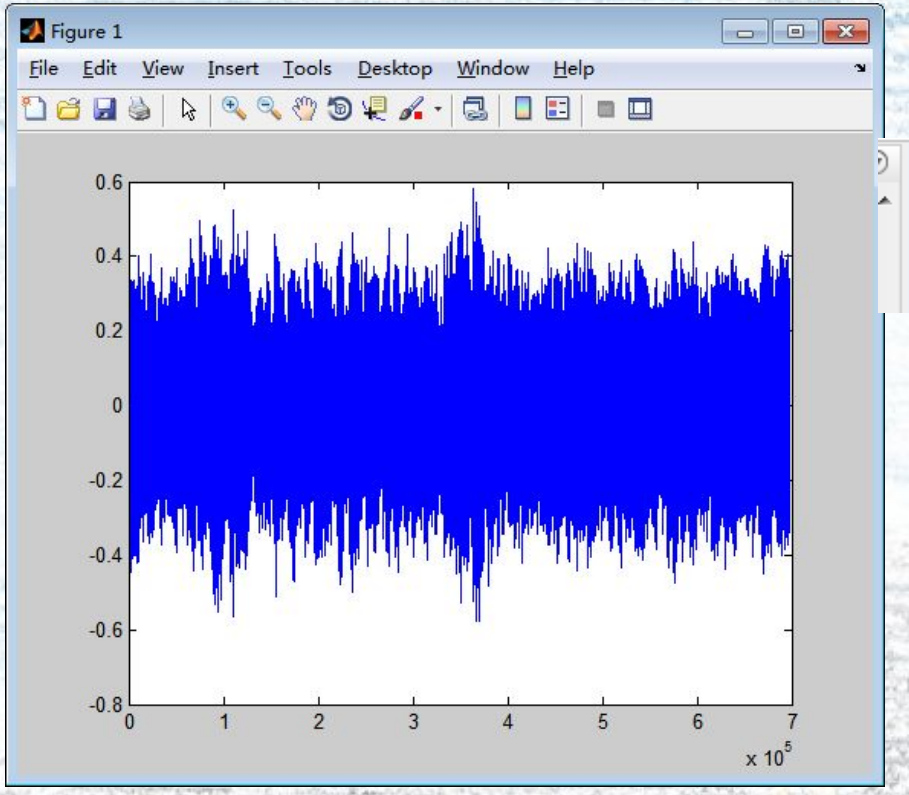
分辨两种声音(噪声和有用信号）的关键在于发现它们的特点，找出它们的不同。

发现规律：一般录音中总是有“空白”部分的，即没人说话的部分，可用来提取关键信息，制做参考输入

微积分的思想，可以近似认为短时间内背景噪声的特征、规律变化不大。因此每隔一段时间可以采一段。

设一个门限值，定时器计时，当持续300ms低于门限时判定为空白，采这一段作为参考，继续采信号，直到下一个满足条件的空白出现

- 读入截取的一段采访的声音：
- **voice=wavread('voice5_new.wav');**
- 用**sound**函数试听一下：
- **sound(voice,44100);**



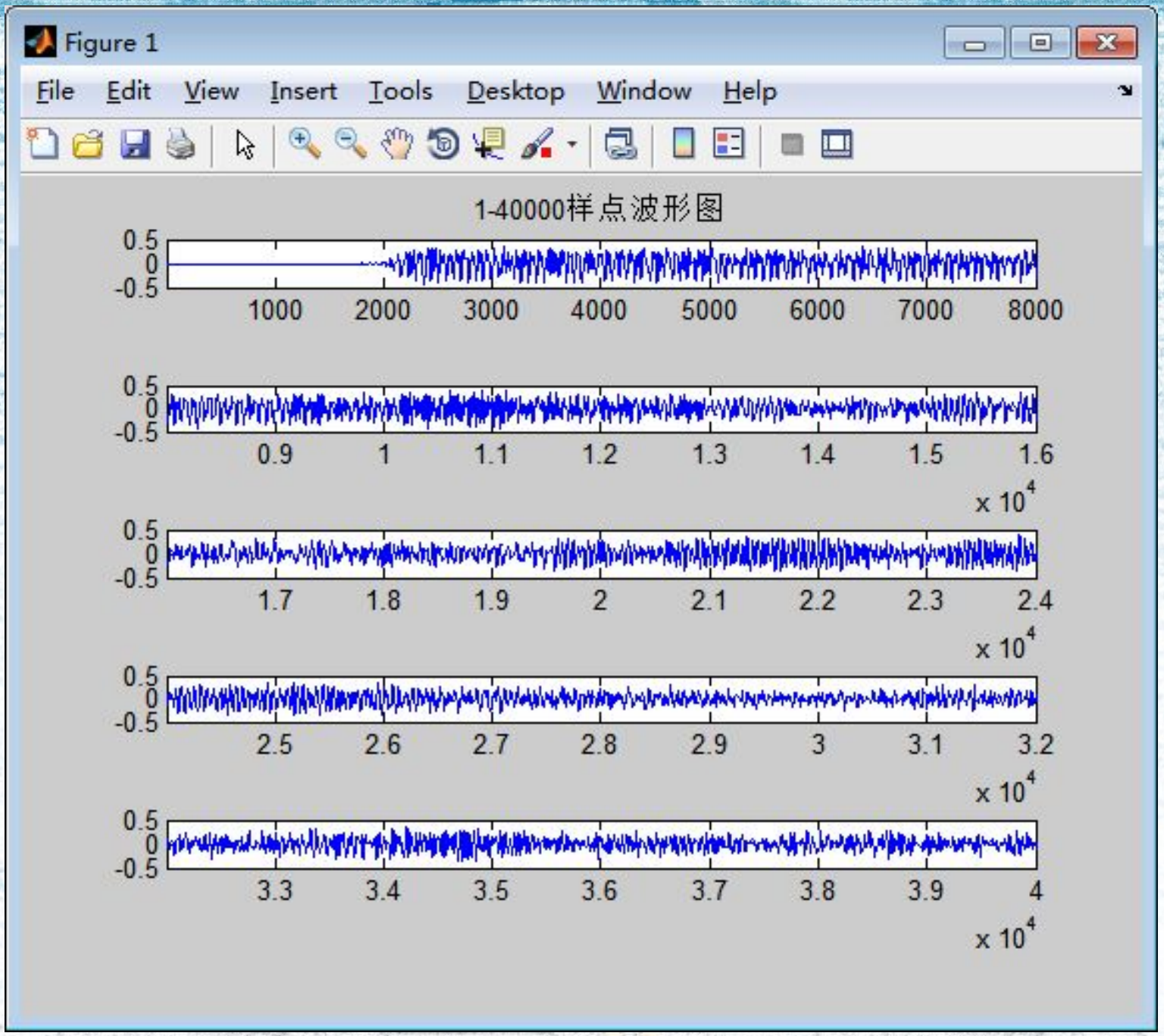


Figure 1
File Edit View Insert Tools Desktop Window Help
1-40000样点波形图
0.5
0
-0.5
1000 2000 3000 4000 5000 6000 7000 8000
0.9 1 1.1 1.2 1.3 1.4 1.5 1.6
x 10^4
1.7 1.8 1.9 2 2.1 2.2 2.3 2.4
x 10^4
2.5 2.6 2.7 2.8 2.9 3 3.1 3.2
x 10^4
3.3 3.4 3.5 3.6 3.7 3.8 3.9 4
x 10^4

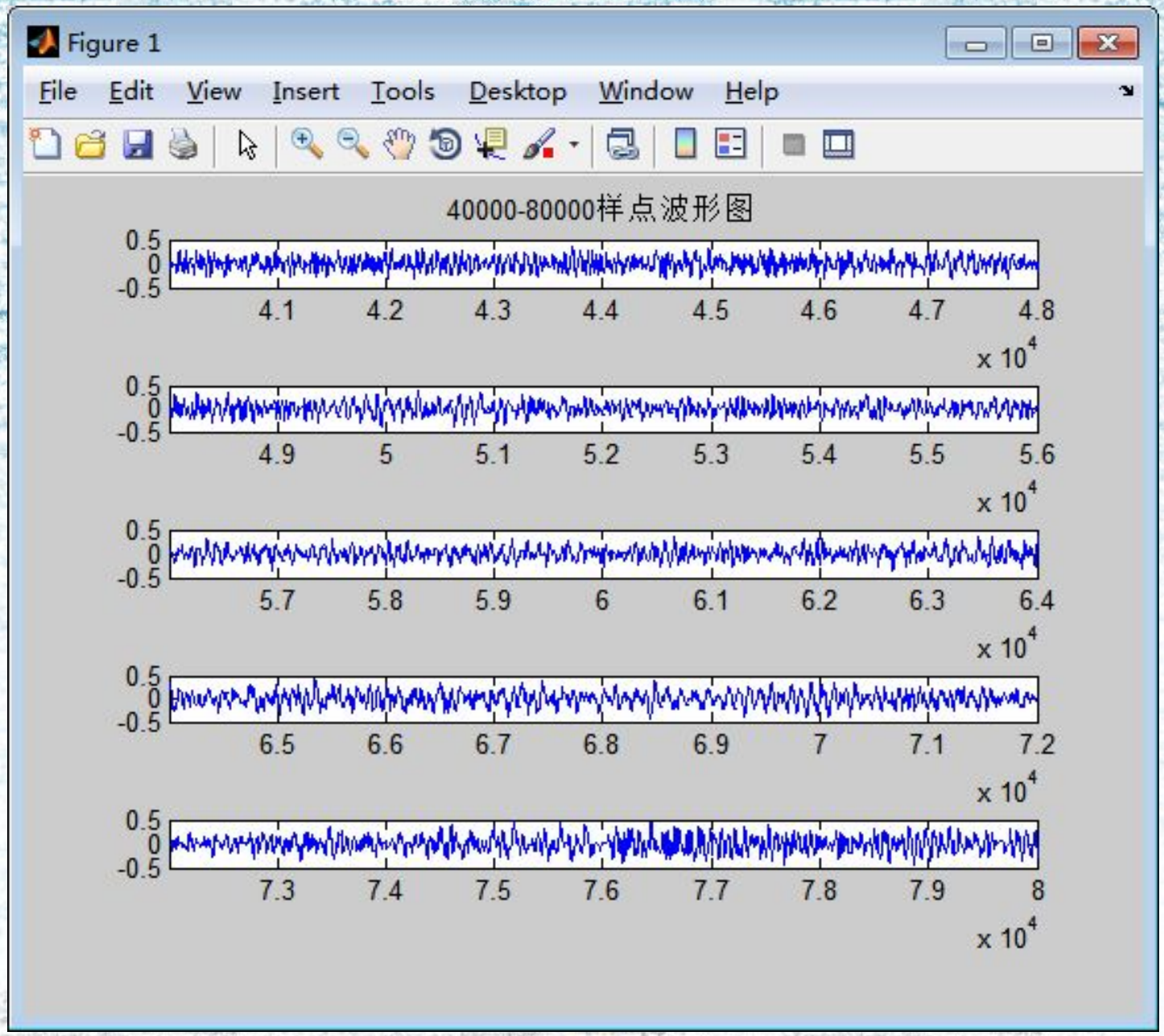


Figure 1
File Edit View Insert Tools Desktop Window Help
40000-80000样点波形图
0.5
0
-0.5
4.1 4.2 4.3 4.4 4.5 4.6 4.7 4.8
x 10^4
4.9 5 5.1 5.2 5.3 5.4 5.5 5.6
x 10^4
5.7 5.8 5.9 6 6.1 6.2 6.3 6.4
x 10^4
6.5 6.6 6.7 6.8 6.9 7 7.1 7.2
x 10^4
7.3 7.4 7.5 7.6 7.7 7.8 7.9 8
x 10^4

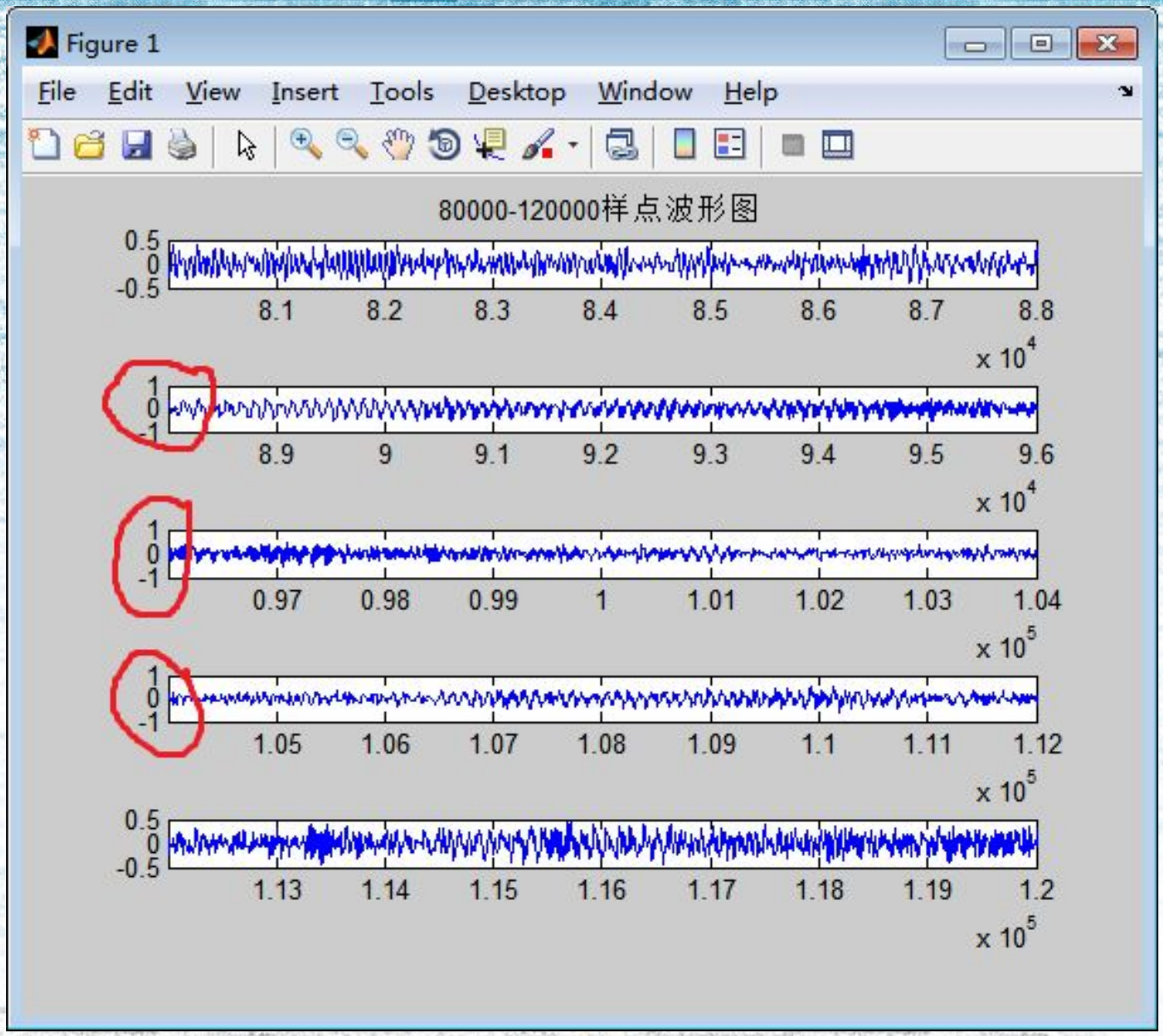


Figure 1
File Edit View Insert Tools Desktop Window Help
80000-120000样点波形图
0.5
0
-0.5
8.1 8.2 8.3 8.4 8.5 8.6 8.7 8.8
x 10^4
1
0
-1
8.9 9 9.1 9.2 9.3 9.4 9.5 9.6
x 10^4
1
0
-1
0.97 0.98 0.99 1 1.01 1.02 1.03 1.04
x 10^5
1
0
-1
1.05 1.06 1.07 1.08 1.09 1.1 1.11 1.12
x 10^5
0.5
0
-0.5
1.13 1.14 1.15 1.16 1.17 1.18 1.19 1.2
x 10^5

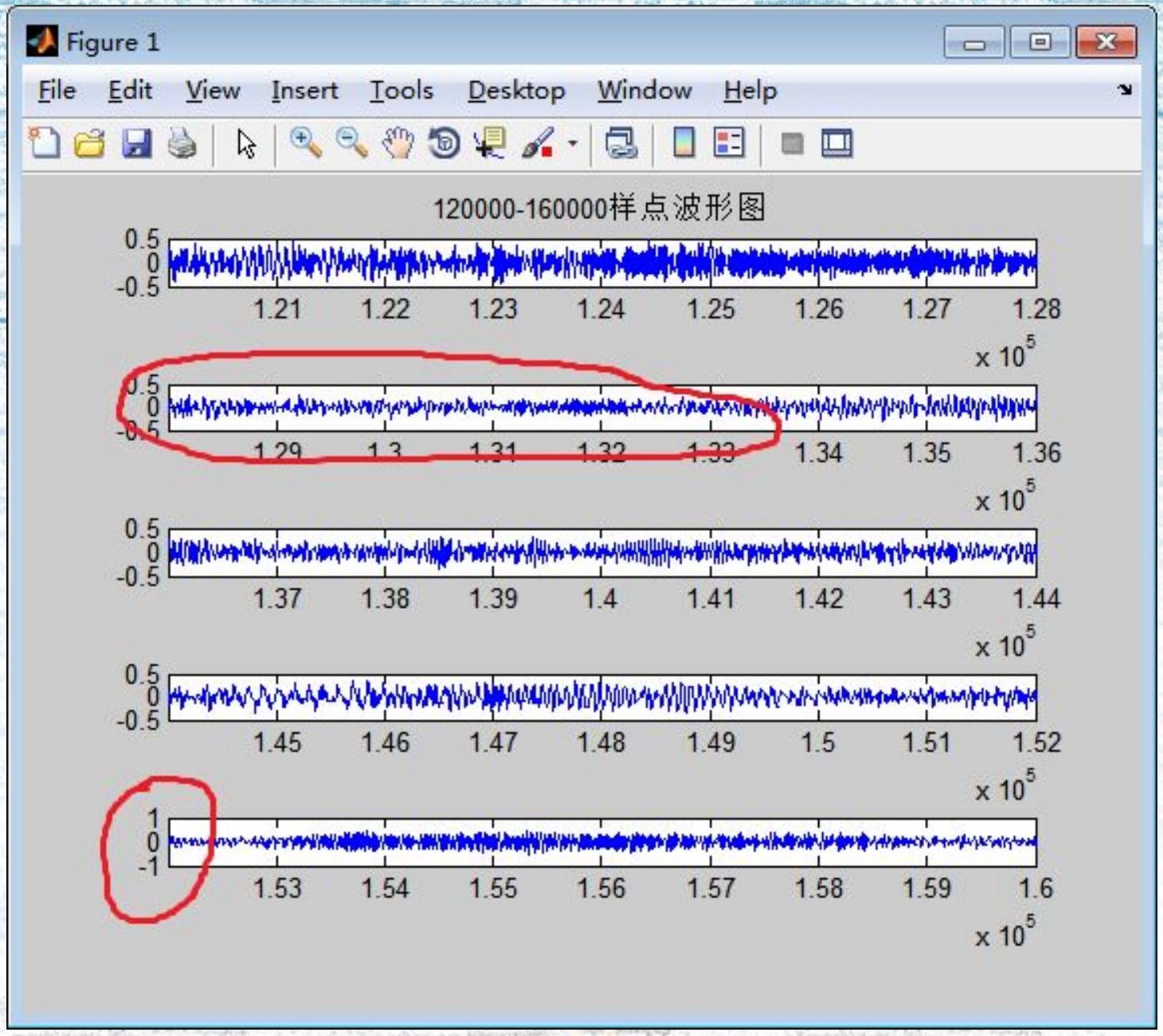
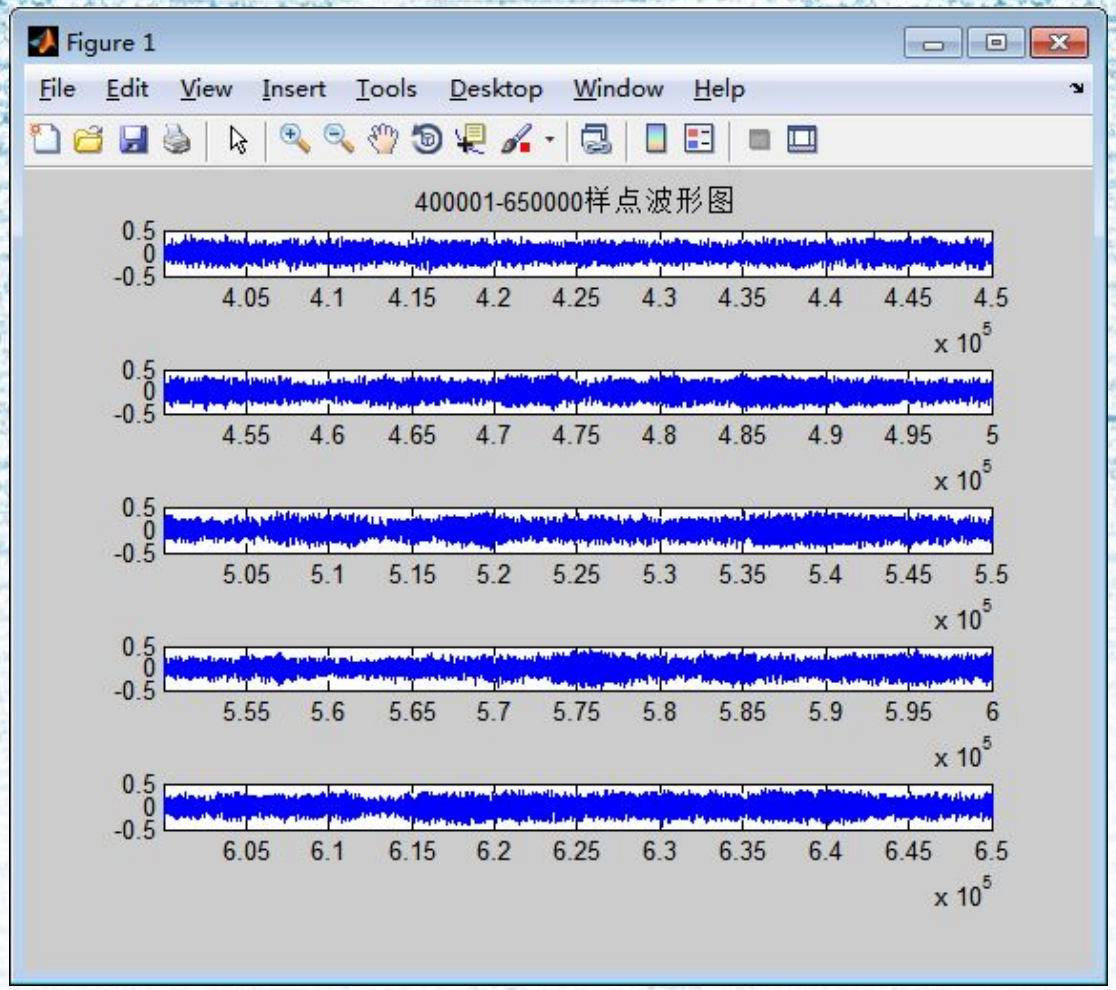


Figure 1
File Edit View Insert Tools Desktop Window Help
120000-160000样点波形图
0.5
0
-0.5
1.21 1.22 1.23 1.24 1.25 1.26 1.27 1.28
x 10^5
0.5
0
-0.5
1.29 1.3 1.31 1.32 1.33 1.34 1.35 1.36
x 10^5
0.5
0
-0.5
1.37 1.38 1.39 1.4 1.41 1.42 1.43 1.44
x 10^5
0.5
0
-0.5
1.45 1.46 1.47 1.48 1.49 1.5 1.51 1.52
x 10^5
1
0
-1
1.53 1.54 1.55 1.56 1.57 1.58 1.59 1.6
x 10^5



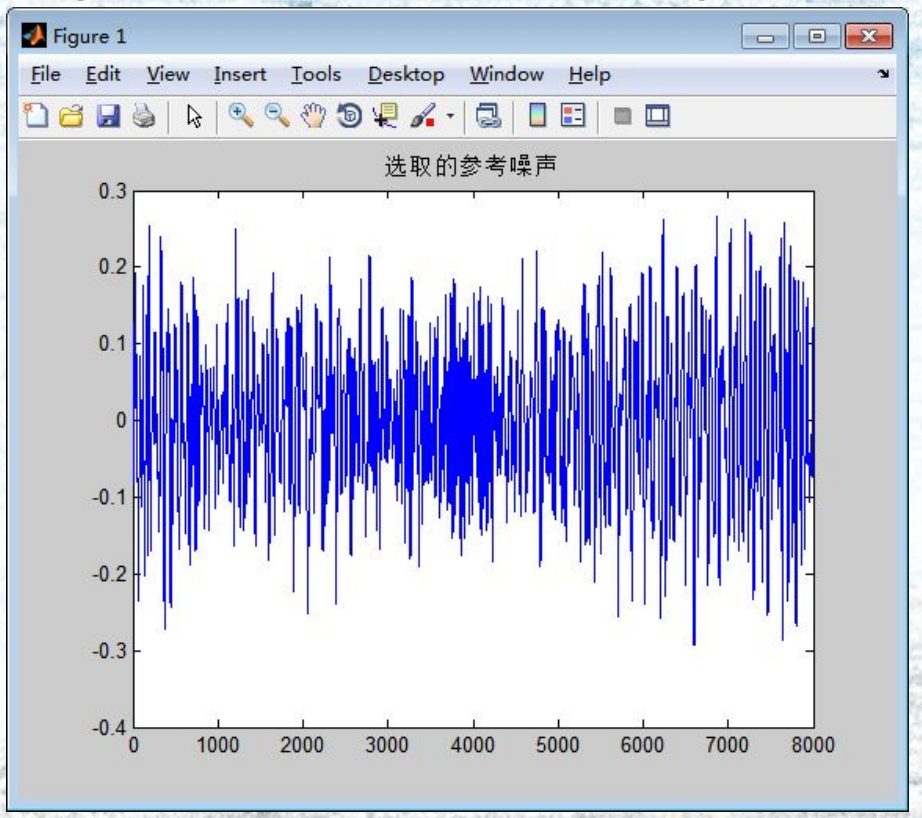
从幅值上来看，人说话的声音，和背景噪声，差距不是非常明显

但人耳听得时候是可以很明显地辨别人说话的声音。说明人耳不仅仅是靠声音的大小来提取有用信息的。。。。有待研究。。。。

还是从幅值上做一下实验。

选取波动最小的一段：

```
noise_ref=wavread('voice5_new.wav',[128
001 136000]);
sound(noise_ref,44100);
```



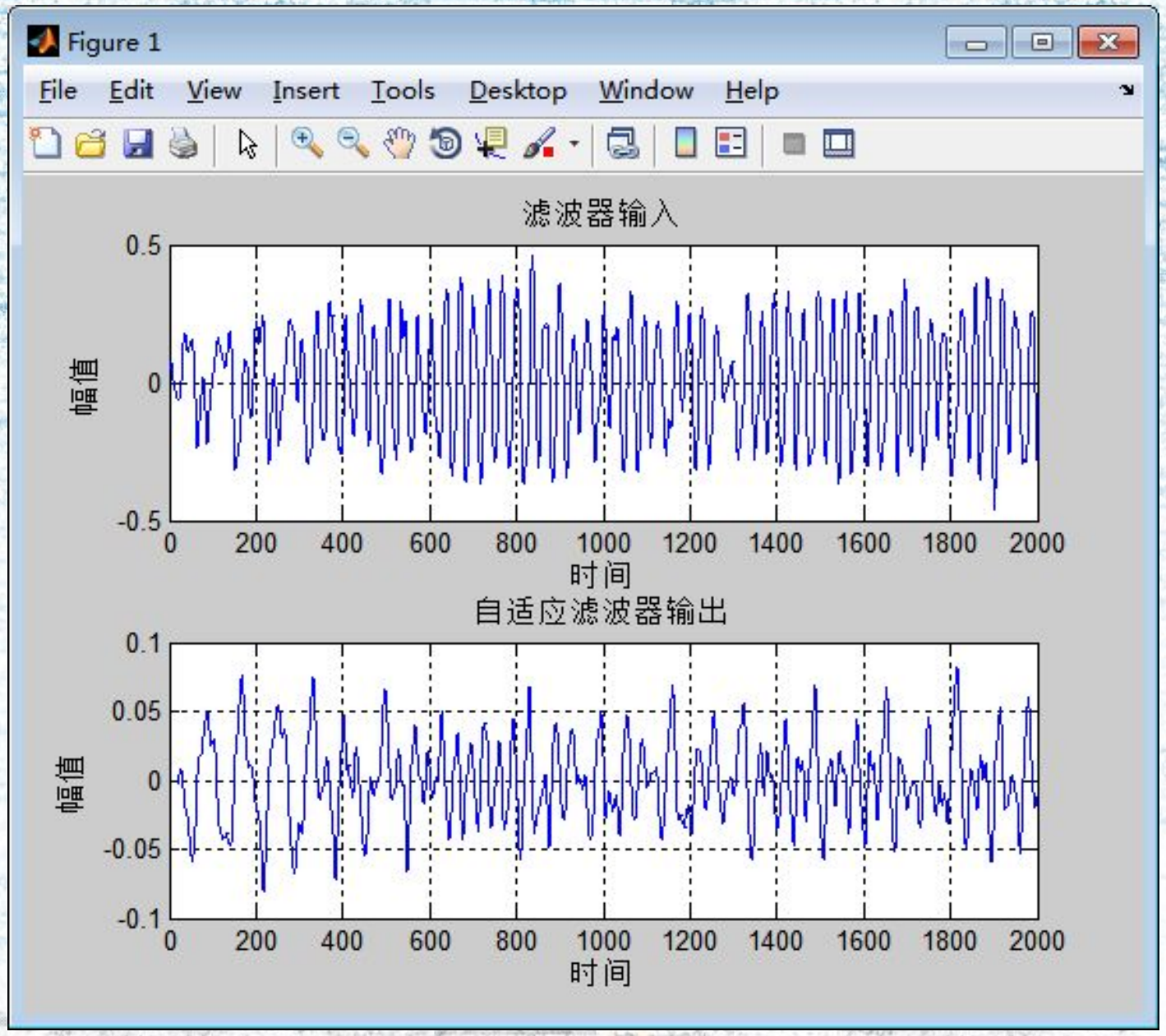


Figure 1
File Edit View Insert Tools Desktop Window Help
滤波器输入
幅值
0.5
0
-0.5
0 200 400 600 800 1000 1200 1400 1600 1800 2000
时间
自适应滤波器输出
幅值
0.1
0.05
0
-0.05
-0.1
0 200 400 600 800 1000 1200 1400 1600 1800 2000
时间

程序主要部分代码

```
t=0:697999;
xn=wavread('voice5_new.wav',[1 698000]);
dn=wavread('voice5_new.wav',[128001 130000]);
ss=zeros(698000,1);
for m=1:349
ss((2000*m-1999):2000*m,1)=dn;
end
 sound(xn,44100,16);
 sound(xn-yn,44100,16);
```

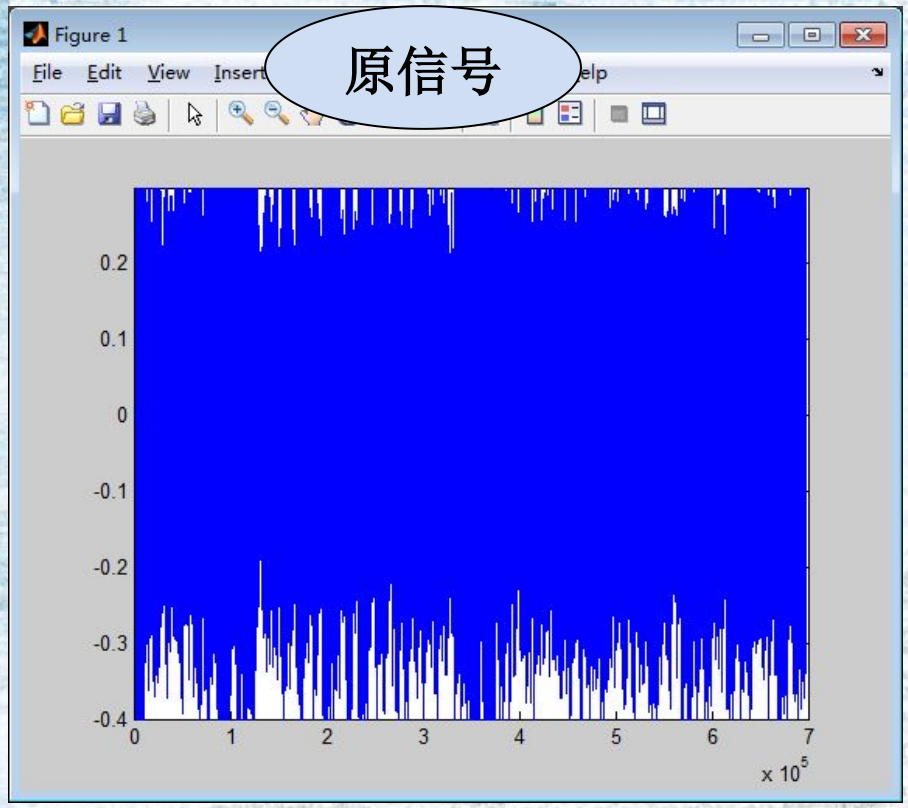
请听 **afterprocess.wav**

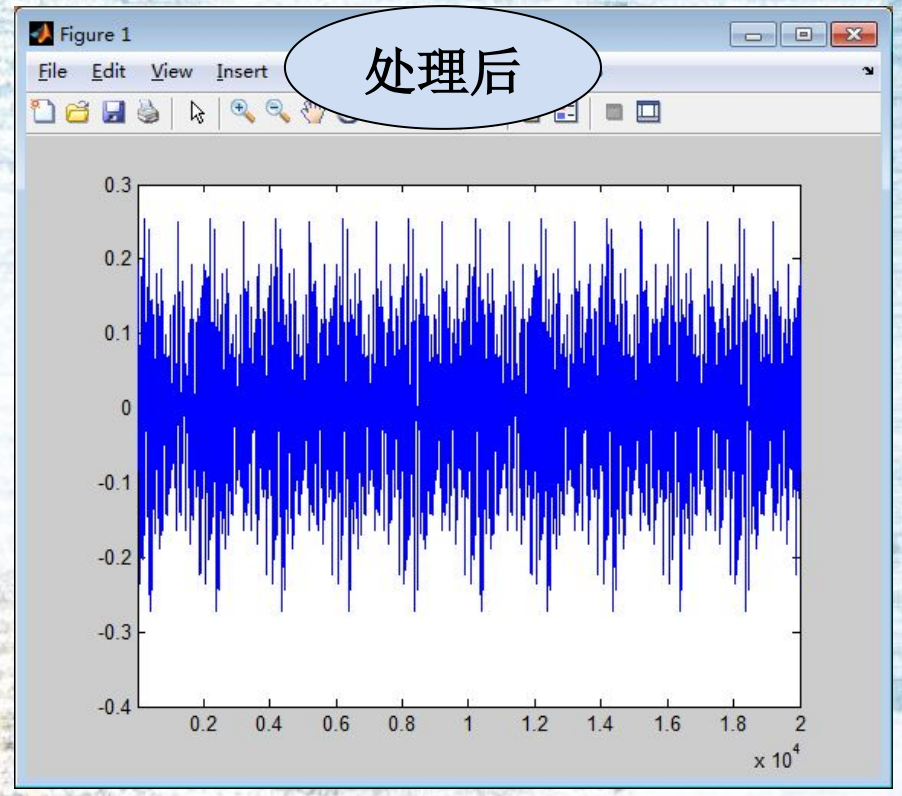
把参考噪声段重复拼接，和原信号同样长。然后作为参考噪声。

由于拼接，存在一个接口的声音去不掉，如下图的尖峰。

处理后的声音虽然噪声小了，但夹杂着马达的声音。

时域图（为了方便比较，我把坐标调成了一样的）





频谱图

原信号

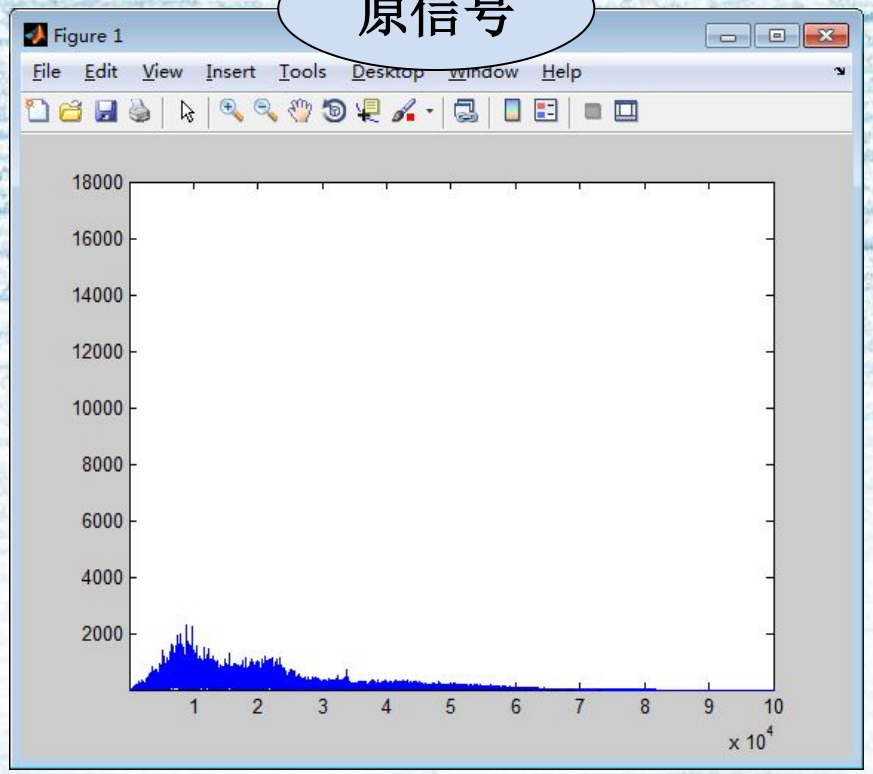


Figure 1
File Edit View Insert Tools Desktop Window Help
18000
16000
14000
12000
10000
8000
6000
4000
2000
1 2 3 4 5 6 7 8 9 10
x 10^4

处理后

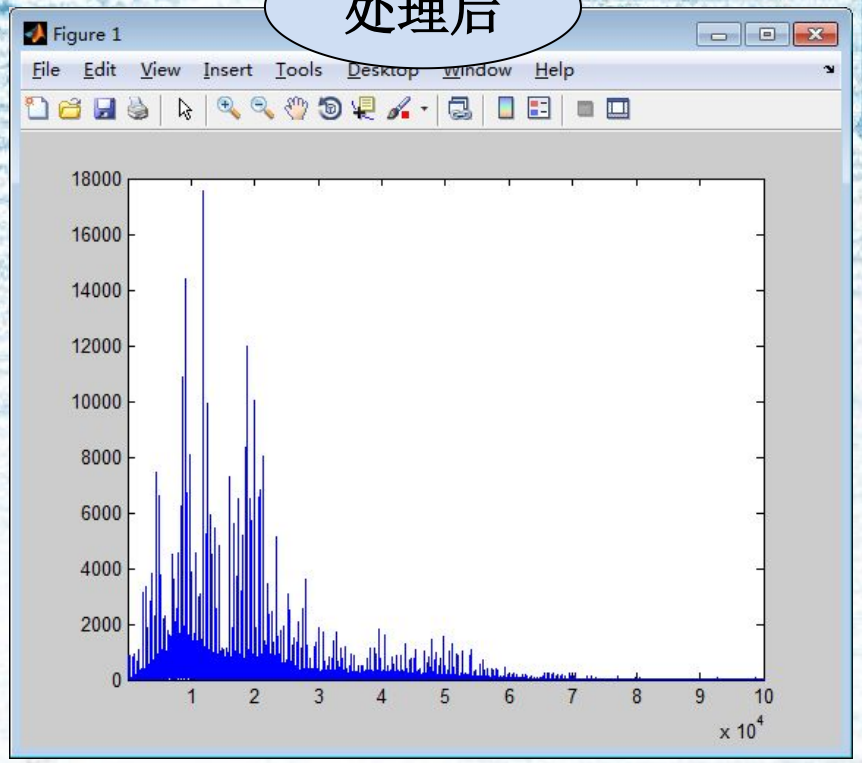
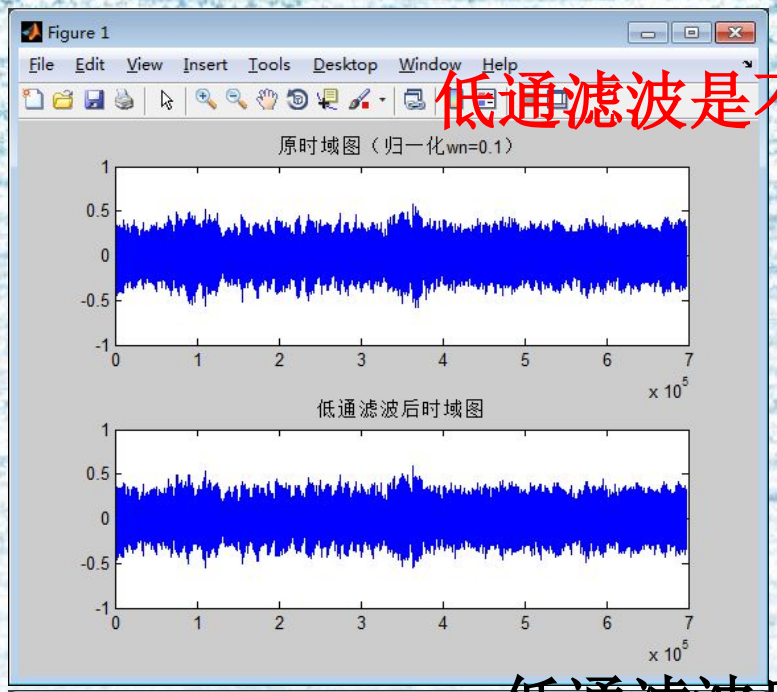
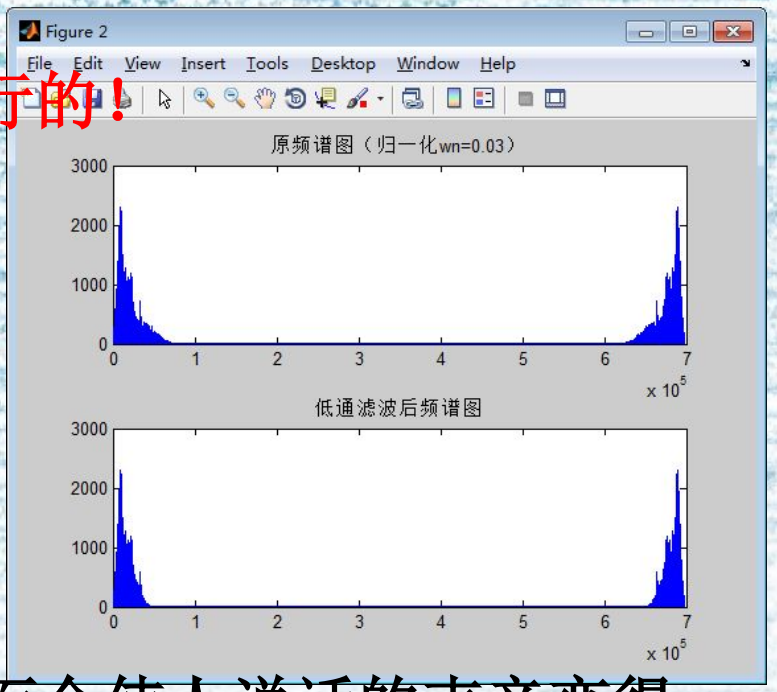
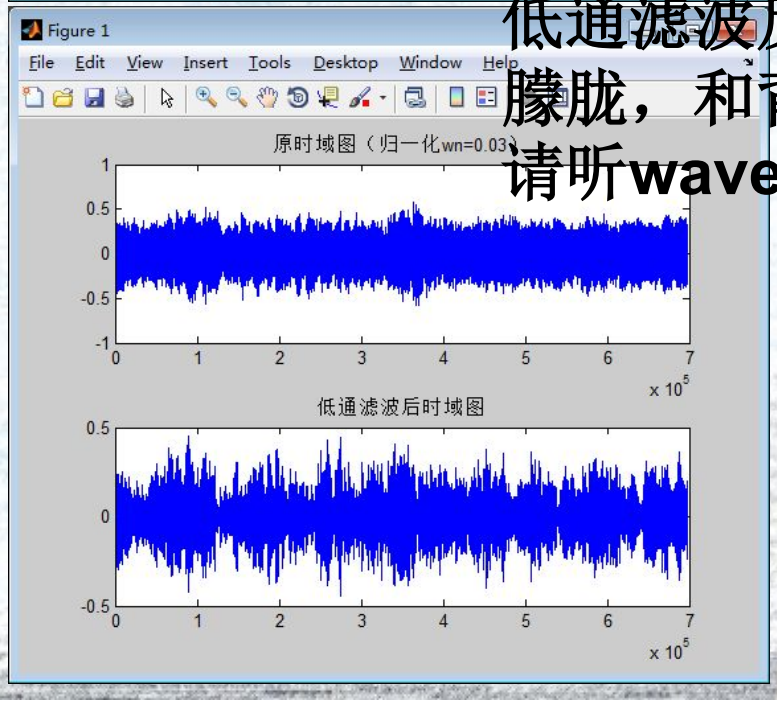
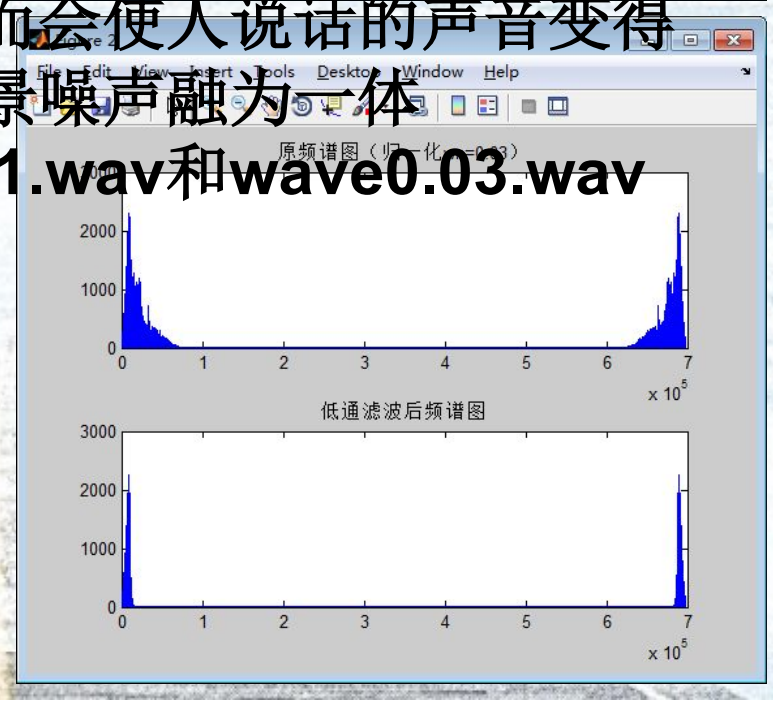


Figure 1
File Edit View Insert Tools Desktop Window Help
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
1 2 3 4 5 6 7 8 9 10
x 10^4









低通滤波是不行的！

低通滤波反而会使人说话的声音变得朦胧，和背景噪声融为一体

请听wave0.1.wav和wave0.03.wav

小结

- 1、自适应滤波的原理
- 2、关键的算法——原理及实现
- 3、matlab的仿真
- 4、影响滤波效果的因素探究
- 5、自适应滤波的应用之一：去噪
- 6、参考噪声的选取——我的思路
- 7、实际解决问题

谢谢收看！