

一、限幅滤波法

1、方法：

根据经验判断两次采样允许的最大偏差值（设为 A）

每次检测到新值时判断：

- a. 如果本次值与上次值之差 $\leq A$ ，则本次值有效
- b. 如果本次值与上次值之差 $> A$ ，则本次值无效，放弃本次值，用上次值代替本次值

2、优点：

能有效克服因偶然因素引起的脉冲干扰

3、缺点

无法抑制那种周期性的干扰

平滑度差

/* A 值根据实际调，Value 有效值，new_Value 当前采样值，程序返回有效的实际值 */

```
#define A 10
```

```
char Value;
```

```
char filter()
```

```
{
```

```
    char new_Value;
```

```
    new_Value = get_ad(); // 获取采样值
```

```
    if( abs(new_Value - Value) > A)
```

```
        return Value;    // abs()取绝对值函数
```

```
    return new_Value;
```

```
}
```

二、中位值滤波法

1、方法：

连续采样 N 次（N 取奇数），把 N 次采样值按大小排列

取中间值为本次有效值

2、优点：

能有效克服因偶然因素引起的波动干扰

对温度、液位的变化缓慢的被测参数有良好的滤波效果

3、缺点：

对流量、速度等快速变化的参数不宜

```
#define N 11
```

```
char filter()
```

```
{  
  
    char value_buf[N];  
  
    char count, i, j, temp;  
  
    for(count = 0; count < N; count++) //获取采样值  
    {  
  
        value_buf[count] = get_ad();  
  
        delay();  
    }  
  
    for(j = 0; j < (N-1); j++)  
    {  
  
        for(i = 0; i < (N-j); i++)  
        {  
  
            if(value_buf[i] > value_buf[i+1])  
            {  
  
                temp = value_buf[i];  
  
                value_buf[i] = value_buf[i+1];  
  
                value_buf[i+1] = temp;  
  
            }  
  
        }  
  
    }  
  
    return value_buf[(N-1)/2];  
}
```

三、算术平均滤波法

1、方法：

连续取 N 个采样值进行算术平均运算

N 值较大时：信号平滑度较高，但灵敏度较低

N 值较小时：信号平滑度较低，但灵敏度较高

N 值的选取：一般流量，N=12；压力：N=4

2、优点：

适用于对一般具有随机干扰的信号进行滤波

这样信号的特点是有一个平均值，信号在某一数值范围附近上下波动

3、缺点：

对于测量速度较慢或要求数据计算速度较快的实时控制不适用

比较浪费 RAM

```
#define N 12
```

```
char filter()
```

```
{  
    int sum = 0;  
    for(count = 0; count < N; count++)  
    {  
        sum += get_ad();  
    }  
    return (char)(sum/N);  
}
```

四、递推平均滤波法

1、方法：

把连续取 N 个采样值看成一个队列

队列的长度固定为 N

每次采样到一个新数据放入队尾，并扔掉原来队首的一次数据.(先进先出原则)

把队列中的 N 个数据进行算术平均运算，就可获得新的滤波结果

N 值的选取：流量，N=12；压力：N=4；液面，N=4 ~ 12；温度，N=1 ~ 4

2、优点：

对周期性干扰有良好的抑制作用，平滑度高

适用于高频振荡的系统

3、缺点：

灵敏度低

对偶然出现的脉冲性干扰的抑制作用较差

不易消除由于脉冲干扰所引起的采样值偏差

不适用于脉冲干扰比较严重的场合

比较浪费 RAM

/* A 值根据实际调，Value 有效值，new_Value 当前采样值，程序返回有效的实际值 */

```
#define A 10
```

```
char Value;
```

```
char filter()
```

```
{
```

```
    char new_Value;
```

```
    new_Value = get_ad(); // 获取采样值
```

```
    if( abs(new_Value - Value) > A)
```

```
        return Value;    // abs()取绝对值函数
```

```
    return new_Value;
```

```
}
```

五、中位值平均滤波法

1、方法：

相当于“中位值滤波法”+“算术平均滤波法”

连续采样 N 个数据，去掉一个最大值和一个最小值

然后计算 N-2 个数据的算术平均值

N 值的选取：3~14

2、优点：

融合了两种滤波法的优点

对于偶然出现的脉冲性干扰，可消除由于脉冲干扰所引起的采样值偏差

3、缺点：

测量速度较慢，和算术平均滤波法一样

比较浪费 RAM

```
char filter()
{
    char count, i, j;

    char Value_buf[N];

    int sum = 0;

    for(count = 0; count < N; count++)
    {
        Value_buf[count] = get_ad();
    }

    for(j = 0; j < (N-1); j++)
    {
        for(i = 0; i < (N-j); i++)
        {
            if(Value_buf[i] > Value_buf[i+1])
            {
                temp = Value_buf[i];
                Value_buf[i] = Value_buf[i+1];
                Value_buf[i+1] = temp;
            }
        }
    }

    for(count = 1; count < N-1; count++)
    {
        sum += Value_buf[count];
    }

    return (char)(sum/(N-2));
}
```

六、限幅平均滤波法

1、方法：

相当于“限幅滤波法”+“递推平均滤波法”

每次采样到的新数据先进行限幅处理，

再送入队列进行递推平均滤波处理

2、优点：

融合了两种滤波法的优点

对于偶然出现的脉冲性干扰，可消除由于脉冲干扰所引起的采样值偏差

3、缺点：

比较浪费 RAM

```
#define A 10

#define N 12

char value, i = 0;

char value_buf[N];

char filter()
{
    char new_value, sum = 0;

    new_value = get_ad();

    if(Abs(new_value - value) < A)
        value_buf[i++] = new_value;

    if(i==N)
        i=0;

    for(count = 0; count < N; count++)
    {
        sum += value_buf[count];
    }

    return (char)(sum/N);
}
```

七、一阶滞后滤波法

1、方法：

取 $a=0\sim 1$

本次滤波结果 = $(1-a)$ 本次采样值 + a 上次滤波结果

2、优点：

对周期性干扰具有良好的抑制作用

适用于波动频率较高的场合

3、缺点：

相位滞后，灵敏度低

滞后程度取决于 a 值大小

不能消除滤波频率高于采样频率的 $1/2$ 的干扰信号

/*为加快程序处理速度，取 $a=0\sim 100$ */

```
#define a 30
```

```
char value;
```

```
char filter()
```

```
{  
    char new_value;  
    new_value = get_ad();  
    return ((100-a)*value + a*new_value);  
}
```

八、加权递推平均滤波法

1、方法：

是对递推平均滤波法的改进，即不同时刻的数据加以不同的权

通常是，越接近现时刻的数据，权取得越大。

给予新采样值的权系数越大，则灵敏度越高，但信号平滑度越低

2、优点：

适用于有较大纯滞后时间常数的对象

和采样周期较短的系统

3、缺点：

对于纯滞后时间常数较小，采样周期较长，变化缓慢的信号
不能迅速反应交易系统当前所受干扰的严重程度，滤波效果差

```
/* coe 数组为加权系数表 */  
  
#define N 12  
  
char code coe[N] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};  
  
char code sum_coe = {1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 + 11 + 12};  
  
char filter()  
{  
    char count;  
  
    char value_buf[N];  
  
    int sum = 0;  
  
    for(count = 0; count < N; count++)  
    {  
        value_buf[count] = get_ad();  
    }  
  
    for(count = 0; count < N; count++)  
    {  
        sum += value_buf[count] * coe[count];  
    }  
  
    return (char)(sum/sum_coe);  
}
```

九、消抖滤波法

1、方法：

设置一个滤波计数器

将每次采样值与当前有效值比较：

如果采样值=当前有效值，则计数器清零

如果采样值>或<当前有效值，则计数器+1，并判断计数器是否>=上限 N(溢出)

如果计数器溢出，则将本次值替换当前有效值，并清计数器

2、优点：

对于变化缓慢的被测参数有较好的滤波效果，

可避免在临界值附近控制器的反复开/关跳动或显示器上数值抖动

3、缺点：

对于快速变化的参数不宜

如果在计数器溢出的那一次采样到的值恰好是干扰值，则会将干扰值当作有效值导入交易系统

```
#define N 12
```

```
char filter()
```

```
{  
    char count = 0, new_value;  
    new_value = get_ad();  
    while(value != new_value)  
    {  
        count++;  
        if(count >= N)  
            return new_value;  
        new_value = get_ad();  
    }  
    return value;  
}
```

十、限幅消抖滤波法

1、方法：

相当于“限幅滤波法”+“消抖滤波法”

先限幅，后消抖

2、优点：

继承了“限幅”和“消抖”的优点

改进了“消抖滤波法”中的某些缺陷，避免将干扰值导入系统

3、缺点：

对于快速变化的参数不宜

```
#define A 10

#define N 12

char value;

char filter()
{
    char new_value, count = 0;

    new_value = get_ad();

    while(value != new_value)
    {
        if(Abs(value - new_value) < A)
        {
            count++;

            if(count >= N)
                return new_value;

            new_value = get_ad();
        }

        return value;
    }
}
```