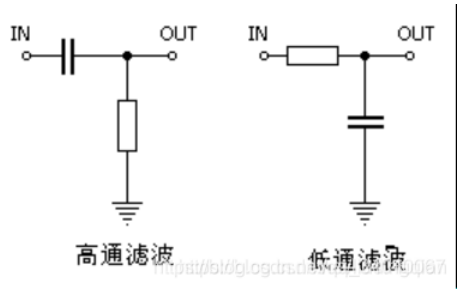
C语言两行代码实现RC高通/低通滤波，拿来即用，不需要移植

最近做心电监测项目，发现信号干扰很严重，图像完全是干扰信号，根本看不出心电信号，公司给了滤波函数，但是高通滤波　不知道什么原因不能用。百度只找到了低通滤波代码(Uo=k*Ui+(1-k)*Uo),k值也没給计算公式，最主要的是没有我需要的高通滤波。数学太菜，搜出来的其他答案大量公式看不懂，符号都不认识，也移植不了。只能自己摸索，花了大量时间，终于搞定高通滤波，把k值计算公式也推导出来了，放出来给需要的童鞋用，免得你们走弯路。

RC滤波电路　图:



1.代码:

```
//k值的计算方法。
//符号:Sr-采样率(sampling rate,次/秒),f-截止频率(Hz),Pi-圆周率(3.14...)
//k=(2*Pi*f)/Sr
typedef struct
{
  float k;//滤波系数
  float lVal;//上次计算值
}rcPara_t;

//低通滤波:
//rcPara-指向滤波参数
//val-采样值
//返回值-滤波结果
float rcLfFiter(rcPara_t *rcPara, float val)
{
    rcPara->lVal=((float)val*rcPara->k+rcPara->lVal*(1-rcPara->k));
    return rcPara->lVal;
}

//高通滤波:
float rcHpFiter(rcPara_t *rcPara, float val)
{
    rcPara->lVal=((float)val*rcPara->k+rcPara->lVal*(1-rcPara->k));
    //    return -(val-rcPara->lVal);//滤波结果
    return (val-rcPara->lVal);//如果直接返回滤波结果，滤波后图像是倒转的，在心电图等一些场合，需要将图像再镜像过来
}
```

2.低通滤波测试代码

```
typedef struct
{
  float freq;//频率
  int ratio;//占空比(%)
  int cycle;//采样率
  int range;//取值范围
  uint32_t count;//数据计数
}fsPara_t;

//正弦信号
#define PI 3.141592653f
float sinSignal(fsPara_t *sgn)
{
  float cnum = (float)sgn->cycle/sgn->freq;
  float unit = 2.0F*PI/(float)cnum;
  float val = 0;
  val += unit*(float)sgn->count++;
  return (sin(val)*sgn->range);
}

//正弦信号频率1Hz，采样率500，幅值100
fsPara_t hsPara = {.freq=1, .cycle=500, .range=100};
rcPara_t rcParaLp = {.k=(2*3.14f*1/500)};
void testRcLfFiterLevel1(void)
```

```
{
  float buff[2000];
  float inValMax=0, outValMax=0;

  for(int i=0; i<2000; i++)
  {
    buff[i]=sinSignal(&hsPara);
    if(buff[i]>inValMax) inValMax=buff[i];
    //printf("%0.2f ", buff[i]);
    buff[i]=rcLfFiter(&rcParaLp, buff[i]);
    if(buff[i]>outValMax && i>1000) outValMax=buff[i];
    //printf("%0.2f\n", buff[i]);

  }
  printf("inValMax:%0.2f outValMax:%0.2f\n", inValMax, outValMax);
}
```

调用testRcLfFiterLevel1();输出结果:inValMax:100.00 outValMax:70.92
70.92/100=0.7092
截止频率下滤波后信号幅值衰减约为0.707(为什么是这个值，自行百度)，很接近了。

信号频率改为0.1：
fsPara_t hsPara = {.freq=0.1, .cycle=500, .range=100};
输出结果:inValMax:100.00 outValMax:99.51
可见低频几乎没有衰减。

信号频率改为10：
fsPara_t hsPara = {.freq=10, .cycle=500, .range=100};
输出结果:inValMax:100.00 outValMax:10.01
高频信号大部分被过滤掉了

多阶低通滤波多次调用即可，测试代码(3阶低通滤波):
ps:多阶滤波截止频率公式f=1/(2*Pi*R*C)已经不适用了，自行推导，我也不不会.
慢慢试也能试出来，幅值衰减到0.707(或功率降到0.5)就是截止

```
//频率1Hz，采样率500，幅值100
fsPara_t hsParaMuti = {.freq=1, .cycle=500, .range=100};
#define RC_F    1.0f
#define RC_K    (2*3.14f*RC_F/500)
rcPara_t rcParaLpMutiL1 = {.k=RC_K};
rcPara_t rcParaLpMutiL2 = {.k=RC_K};
rcPara_t rcParaLpMutiL3 = {.k=RC_K};
void testRcLfFiterLevel3(void)
{
  float buff[2000];
  float inValMax=0, outValMax=0;

  for(int i=0; i<2000; i++)
  {
    buff[i]=sinSignal(&hsParaMuti);
    if(buff[i]>inValMax) inValMax=buff[i];
    //printf("%0.2f ", buff[i]);
    buff[i]=rcLfFiter(&rcParaLpMutiL1, buff[i]);
    buff[i]=rcLfFiter(&rcParaLpMutiL2, buff[i]);
    buff[i]=rcLfFiter(&rcParaLpMutiL3, buff[i]);
    if(buff[i]>outValMax && i>1000) outValMax=buff[i];
    //printf("%0.2f\n", buff[i]);

  }
  printf("inValMax:%0.2f outValMax:%0.2f\n", inValMax, outValMax);
}
```

调用testRcLfFiterLevel3();输出结果:inValMax:100.00 outValMax:35.66
35.66/100=0.3566
3阶滤波幅值衰减为0.707^3=0.35339…，也很接近。

3.高通滤波测试代码

```
fsPara_t lsPara = {.freq=1, .cycle=500, .range=100};
rcPara_t rcParaHp = {.k=(2*3.14f*1/500)};
void testRcHpFiterLevel1(void)
{
  float buff[2000];
  float inValMax=0, outValMax=0;

  for(int i=0; i<2000; i++)
  {
    buff[i]=sinSignal(&lsPara);
    if(buff[i]>inValMax) inValMax=buff[i];
```

```
    //printf("%0.2f ", buff[i]);
    buff[i]=rcHpFiter(&rcParaHp, buff[i]);
    if(buff[i]>outValMax && i>1000) outValMax=buff[i];
    //printf("%0.2f\n", buff[i]);

  }
  printf("inValMax:%0.2f outValMax:%0.2f\n", inValMax, outValMax);
}
```

调用testRcHpFiterLevel1();输出结果:inValMax:100.00 outValMax:70.06，
很接近0.707。

信号频率改为0.1，输出结果:inValMax:100.00 outValMax:3.94
低频几乎被过滤掉了.

信号频率改为10，输出结果:inValMax:100.00 outValMax:98.80
高频几乎不受影响.

4.接下来是低通滤波公式推导(高通滤波公式可以用相同的思路推导出来)
PS:非专业分析，仅供参考，如有错误请指正。

符号意义：
Q-电荷量(库伦)，C-电容(F)，R-电阻(欧姆)，U-电压(V)，Sr-采样率(sampling rate, 次/秒)
f-截止频率(Hz)，Pi-圆周率(3.14…), t-时间

根据RC低通电路图可知，输出电压等于电容电压
如果采样电压与电容电压不一致，电容就会充/放电，
假设电容充电后电压增加Ur（放电则Ur为负数）：

符号:Uo-输出电压，U-电容电压, Ui-输入电压
1 Uo=U+Ur

因为C=Q/U, 所以U=Q/C
根据U=Q/C, Q=I*t，I=U/R, t=1/Sr得：
2 Ur=(Ui-U)/(R*C*Sr)

代入1得：
3 Uo=U+(Ui-U)/(R*C*Sr)

由于我们是软件滤波，不需要真正的电容电阻，所以令k=1/(R*C*Sr)，代入得：
Uo=U+(Ui-U)*k 简化：
4 Uo=k*Ui+(1-k)*U

到这里已经得出滤波公式，如何通过截止频率得出k值呢

根据截止频率公式f=1/(2*Pi*R*C)得：
RC=1/(2*Pi*f)
代入k=1/(R*C*Sr)得：
5 k=(2*Pi*f)/Sr