Heart Beat Peaks Detection

We detect peaks of raw heart data by three steps:

1. **Normalize the max value on every peak to 1.0.**
   There is always large difference among magnitudes of different peaks. Thus, we normalize them for an easier detection.
2. **Detect peaks by two thresholds**
   One threshold indicates the starting of peak and the other threshold judges the end of this peak.
3. **BPM estiation**
   We estimate the heart beat rate by last five peaks. We record their time interval and then calculate the BPM.

All the source code for this method are as follows: (The code is implemented by C language)

```
sample = r.hrv;          //This is the raw heart beat data
sclTmp = r.scl;          //This is the raw skin conductance data

if (abs(sample-sample_last)<5)  //this threshold can remove the sudden change of raw data
{
	/************** auto-gain ***************/
	//We want to normalize the max magnitude to 1.0 so that we can find peak easier
	if ( sample > peakAG )
		peakAG = _attack*sample;
	else
		peakAG = _decay*peakAG;

	gain = _attack/peakAG;
	sampleAG = gain*sample;

	/*************** peak detect ***************/
	//we set two bounds to find the peak, lower_bound and upper_bound
	if ( sampleAG >= lower_bound )    // peak starts
		near_peak = 1;

	if ( (near_peak==1) && ( sampleAG < upper_bound) ) // peak ends
	{
		near_peak = 0;
		if (beats >= 5)  // here we store last five peaks' time to estimate the
		                 // heart beat rate (BPM)
		{
			tc = clock();    //current time
			index = beats%5;
			//Heart beat rate is the number of peaks in 1 minute. We use average
			//time of recent 5 peaks to give an estimation. The average time is
			//            t_avg = (tc-tl[index])/CLOCKS_PER_SEC/5
			//Then we estimate how many beat peaks in 1 minute (60 s):
			//         BPM = 60/t_avg = 60*5*CLOCKS_PER_SEC/(tc-tl[index])
			BPM = (int)(300*CLOCKS_PER_SEC/(tc-tl[index]));
			tl[index] = tc;  //store current time into time array

			BPMData = BPM;       //Put BPM estimation to the buffer for reading
```

```
            }
            else
            {
                  tl[beats] = clock();  //directly save the first 5 peaks's time
            }
            beats++;
      }

      sample_last = sample;        //update sample_last for removing sudden change
}
```

Appendix. (Initial parameter setting)

```
peakAG = 0;
_attack = 0.9875;
_decay = 0.992;
lower_bound = 0.9975;
upper_bound = 0.99;
near_peak = 0;
beats = 0;
sample_last = 1.5;
```