Using heart functions

Maximilien Chaumon

The latest version of this manual and the MATLAB functions it refers to can be found at github.com/dnacombo/heart_functions

Using heart_peak_detect.m

All that is necessary in the setup is to include the path to the two functions in the MATLAB path.

```
%% Setup

% add the folder with heart_SV_calc.m and heart_peak_detect.m to the path
% (you need to edit this)
addpath(cdhome('heart'))
```

Data can be read using FieldTrip's ft_preprocessing function

```
%% Read the data using fieldtrip
% Simply read a dataset with a cardiac data.

cfg                     = [];
cfg.dataset             = 'MERGE_CUT_pilote1_damier_EBI_12_5_kHz_tsss.fif';
cfg.channel             = {'EBI_-_Magnitude' 'ECG1' 'HeartSound'};
data_raw                = ft_preprocessing(cfg);
```

Detecting heart peaks can be done as follows. All what is needed is an ECG channel (here 'ECG1').

```
%% Detecting heart peaks
% If there is an ECG channel, we can detect heart peaks P Q R S T.
% Everything is based on detecting the R peaks.

cfg = [];
cfg.channel = 'ECG1';

[HeartBeats] = heart_peak_detect(cfg,data_raw);
```

The function can also be called with simpler (legacy, not recommended) method:

```
ECG = data_raw.trial{1}(2,:);
fs = data.fsample;

[HeartBeats] = heart_peak_detect(ECG,fs);
```

The algorithm of the function goes as follows:

The signal is first high and low pass filtered (default 1-100 Hz). The square of the z-scored ECG is computed and a first detection is performed as the peaks passing the cfg.thresh (default 10). Not all R peaks need to be selected at this step. Just enough to create a template heart beat ECG (HB) is necessary. If cfg.plotthresh is true, then a figure is shown, allowing the user to edit the threshold. Then a template HB is computed (shown in a figure if cfg.plotbeat) and convolved with the whole ECG time series. The resulting convolution is normalized to have a maximum of 1 and beats are taken as peaks above cfg.corthresh (cfg.corthresh).

In both steps, a minimum distance between beats of cfg.mindist is enforced.

Other peaks are found based on each R peak. Q is the minimum within 50 ms before R, S is the minimum within 100 ms after R, and T is the maximum between the S peak and a maximum QT interval of 420 ms (a rough standard...).

If cfg.plotcorr is true, then a figure with all heart beats is shown. The user is asked whether or not they want to change the threshold of the correlation. In last resort, outlier heart beats (based on interbeat interval) can be remodeled.

Using heart_SV_calc.m

Need to write explanatory text here...

```
%% Stroke volume
% Simplest call without any feedback
cfg = [];
cfg.interactive = 'no';
cfg.L = L;

[HeartBeats] = heart_SV_calc(cfg,data_raw);

%% A simple plot
% A histogram of stroke volumes extracted at each heart beat

% Stroke volume in mL
StrokeVol = [HeartBeats.SV];

figure;
hist(StrokeVol,20)
xlabel('Stroke volume (mL)')
ylabel('Number of strokes');

%% Use feedback to remove outliers

cfg = [];
cfg.L = L;

[HeartBeats] = heart_SV_calc(cfg,data_raw);
```

Text is especially needed here!

```
%% A new simple plot
% A histogram of stroke volumes extracted at each heart beat

StrokeVol = [HeartBeats.SV];

figure;
hist(StrokeVol,20)
xlabel('Stroke volume (mL)')
ylabel('Number of strokes');
%% Compute other useful values
% Heart rate, IBI, Cardiac output...

% R peaks in seconds
R = [HeartBeats.R_time];

% Average heart rate in pulse/min
HR = numel(R) / (R(end) - R(1)) * 60;

% IBI in seconds
IBI = diff(R);

% Instantaneous Heart Rate at each HB
IHR = 1./IBI.*60;

% Cardiac output in L/min
CO = mean(StrokeVol) * HR / 1000;
```

```
%% Without preloading the data

cfg = [];
cfg.dataset = file;
cfg.L = L;

[HeartBeats] = heart_SV_calc(cfg);
```