PhysIO Toolbox

QuickStart Manual

Author: Lars Kasper, Institute for Biomedical Engineering, University of Zurich and ETH Zurich
kasper@biomed.ee.ethz.ch

Development: 2008-2014
Last Change: May 3rd, 2014
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table of Contents</td>
<td>2</td>
</tr>
<tr>
<td>Purpose</td>
<td>4</td>
</tr>
<tr>
<td>One-Page Quickstart – SPM Toolbox</td>
<td>5</td>
</tr>
<tr>
<td>Quickstart – Matlab script (command line)</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Setting sqpar and thresh-parameters</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Description of Variables: the physio-structure</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Example (main_ECG3T.m)</td>
<td>10</td>
</tr>
<tr>
<td>Step-by-Step Guide</td>
<td>14</td>
</tr>
<tr>
<td>3.1 Interpreting the Output Figures</td>
<td>15</td>
</tr>
<tr>
<td>Citing this work</td>
<td>17</td>
</tr>
<tr>
<td>Example Datasets</td>
<td>18</td>
</tr>
<tr>
<td>4.1 Philips</td>
<td>18</td>
</tr>
<tr>
<td>4.1.1 ECG3T</td>
<td>18</td>
</tr>
<tr>
<td>4.1.2 ECG7T</td>
<td>20</td>
</tr>
<tr>
<td>4.1.3 Pulse Oximeter 3T</td>
<td>22</td>
</tr>
<tr>
<td>4.1.4 ECG3T_TRIGGER</td>
<td>23</td>
</tr>
<tr>
<td>4.2 GE</td>
<td>25</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Pulse Oximeter 3T</td>
</tr>
<tr>
<td>4.3</td>
<td>Siemens</td>
</tr>
<tr>
<td>4.3.1</td>
<td>ECG 3T</td>
</tr>
<tr>
<td>4.4</td>
<td></td>
</tr>
<tr>
<td>4.4</td>
<td></td>
</tr>
<tr>
<td>5.1</td>
<td>log_files</td>
</tr>
<tr>
<td>5.2</td>
<td>sqpar</td>
</tr>
<tr>
<td>5.3</td>
<td>thresh</td>
</tr>
<tr>
<td>5.3.1</td>
<td>thresh.scan_timing</td>
</tr>
<tr>
<td>5.3.2</td>
<td>thresh.cardiac</td>
</tr>
<tr>
<td>5.4</td>
<td>model</td>
</tr>
<tr>
<td>5.5</td>
<td>verbose</td>
</tr>
<tr>
<td>7.2</td>
<td></td>
</tr>
<tr>
<td>TODO/Feature requests</td>
<td></td>
</tr>
</tbody>
</table>
This toolbox provides model-based physiological noise correction of fMRI data using peripheral measures of respiration and cardiac pulsation. It incorporates noise models of cardiac/respiratory phase (RETROICOR, Glover et al. 2000), as well as heart rate variability and respiratory volume per time (cardiac response function, Chang et. al, 2009, respiratory response function, Birn et al. 2006). The toolbox is usable via the SPM batch editor, performs automatic pre-processing of noisy peripheral data and outputs nuisance regressor files directly suitable for SPM ("multiple_regressors.txt").
1. Copy the PhysIO Toolbox code folder to the toolbox folder of spm (Optional) Rename the folder to something meaningful, e.g. PhysIO (see Figure 1).
2. (Re-)Start SPM (spm fmri) and the Batch editor.
3. The PhysIO Toolbox should now occur under SPM -> Tools -> TAPAS PhysIO Toolbox
4. change directory (!) to examples/Philips/ECG3T-folder and load an example spm_job file into the batch editor, e.g : example_spm_job_ECG3T.m
5. Press play!
Note: For further information on the PhysIO Toolbox, consult the handbook or see below.
**QUICKSTART – MATLAB SCRIPT (COMMAND LINE)**

Adapt main_ECG3T.m (or one of the other main_* example files), especially the sequence parameter of the sqpar structure variable and the gradient/ECG/breathing threshold parameters in the thresh structure variable. You may set the parameters of each variable either separately, i.e. via sqpar.Nslices = 30; sqpar.Nscans = 320; or call the struct-command in Matlab to set them at once (see Example).

### 2.1 Setting sqpar and thresh-parameters

**sqpar**

- is a structure holding all relevant timing parameters of your MR sequence
- is needed to time the physiological confound regressors correctly (see chapter 4, Input structures)
- In an ideal world, this is the only structure to be changed in the main_{PPU/ECG3T/ECG7T}.m-example files to run your own logfiles
- In practice, both scan timing and physiological signal need some preprocessing determined by the thresh-structure.
  - The need for this preprocessing should be assessed scrutinizing the output plots of the toolbox.

**thresh.scan_timing**

- determines which sampling points of the physiological logfile will be used for the confound regressor creation.
- can be left empty (=[]) to rely on nominal sequence timing as specified in sqpar, counting volume-TRs
- set for Philips logfiles, if slice/volume scan onsets shall be determined from the logged MR gradient time-course which in the Philips SCANPHYSLOG-file.
Unfortunately, there is no direct acquisition trigger event logged by Philips, so we have to resort to this workaround finding patterns in the gradient time course relating to slice or volume onsets.

- multiple options for detection and count of scan events (see chapter 4, Input structures, for details)
  - from start or end of the log file
  - detecting different gradient amplitudes or temporal spacing for first and other slices of a volume
  - Figure 2 shall give a visualization of these parameters and shows the example output of ECG_3T (for thresh.scan_timing.vol) and ECG_7T (for thresh.scan_timing.vol_spacing):

**Figure 2: Raw Timecourses of Physiological Logfile**

![Figure 2: Raw Timecourses of Physiological Logfile](image)

- Figure 2. thresh.vol and thresh.vol_spacing: Visualisation when which gradient thresholding shall be used and in which figures the corresponding plots are found.
2.2 Description of Variables: the physio-structure

All parameters are occurring in the example files collected in the physio-structure, which can be created using the command

```matlab
physio = physio_new();
```

In the body of this function, each parameter is documented with its usage and possible values. Additionally, `physio_new` can be called with template-names, i.e. typical use cases, e.g. when a manual correction of missed ECG pulses is desired

```matlab
physio = physio_new();
```
2.3 Example (main_ECG3T.m)

This example can be found in examples/main_ECG3T.m See the examples section for details concerning the data.
%% 0. Put code directory into path; for some options, SPM should also be in the path

pathRETROICORcode = fullfile(fileparts(mfilename('fullpath')),'../../../code');
addpath(genpath(pathRETROICORcode));

physio = physio_new();
log_files = physio.log_files;
thresh = physio.thresh;
sqpar = physio.sqpar;
model = physio.model;
verbose = physio.verbose;

%% 1. Define Input Files

log_files.vendor = 'Philips';
log_files.cardiac = 'SCANPHYSLOG.log';
log_files.respiration = 'SCANPHYSLOG.log';

%% 2. Define Nominal Sequence Parameter (Scan Timing)

sqpar.Nslices = 37;
sqpar.NslicesPerBeat = 37;
sqpar.TR = 2.50;
sqpar.Ndummies = 3;
sqpar.Nscans = 495;
sqpar.onset_slice = 19;
 sqpar.Nprep = [] ; % set to >=0 to count scans and dummy volumes from beginning of run, i.e.
logfile,, % includes counting of preparation gradients
sqpar.TimeSliceToSlice = sqpar.TR / sqpar.Nslices;

%% 3. Define Gradient Thresholds to Infer Gradient Timing (Philips only)
% 3.1. Determine volume start solely by marking every Nslices-th scan slice
% event as volume event

use_gradient_log_for_timing = true; % true or false
if use_gradient_log_for_timing
    thresh.scan_timing.grad_direction = 'y';
    thresh.scan_timing.zero = 1700;
    thresh.scan_timing.slice = 1800;
    thresh.scan_timing.vol = [];
    % leave [], if unused; set value >= .slice, if volume % start gradients are higher than slice gradients
    thresh.scan_timing.vol_spacing = [];
    % leave [], if unused; set to e.g. 50e-3 (seconds), if there is a time gap between last slice of a volume & first slice of the next
else
    thresh.scan_timing = [];
end

%% 4. Define which Cardiac Data Shall be Used
thresh.cardiac.modality = 'ECG';
thresh.cardiac.initial_cpulse_select.method = 'load_from_logfile';
thresh.cardiac.posthoc_cpulse_select.method = 'off';

%% 5. Order of RETROICOR-expansions for cardiac, respiratory and %% interaction terms. Option to orthogonalise regressors
model.type = 'RETROICOR';
model.order = struct('c',3,'r',4,'cr',1, 'orthogonalise', 'none');
model.input_other_multiple_regressors = 'rp_fMRI.txt'; % either .txt-file or .mat-file (saves variable R)
model.output_multiple_regressors = 'multiple_regressors.txt';

%% 6. Output Figures to be generated
verbose.level = 2; % 0 = none; 1 = main plots (default); 2 = debugging plots, for setting up new study; 3 = all plots
verbose.fig_output_file = 'PhysIO_output.ps';

%% 7. Run the main script with defined parameters
physio.log_files = log_files;
physio.thresh = thresh;
physio.sqpar = sqpar;
physio.model = model;
physio.verbose = verbose;

[physio_out, R, ons_secs] = physio_main_create_regressors(physio);
By answering the following structured questions, you will be able to choose all options of the physIO toolbox according to the specific properties of your physiological dataset and modeling requirements. They are ordered by the general workflow of the physIO toolbox as depicted in Figure 3.

**Figure 3: General Workflow of the physIO toolbox**

- **Read logfiles**
  - Which vendor is used?
    - Philips
    - GE
    - Siemens
  - How shall the timing of the scan triggers and physiological logfile be synchronized?
    - Using a nominal timing
    - Using the gradient time-course (Philips only)
- (Using the gradient-induced peaks in the unfiltered ECG)
- Preprocess physiological data
  - ECG or PPU?
  - heart beat peaks loaded from logfile (as detected) or initial re-detection
  - post-hoc manual labeling of missing heart beats?

### 3.1 Interpreting the Output Figures

The following figures give an overview of the visual output of the toolbox for correct physiological logfile data (from Philips).

![Figure 4. Reference output of scan-timing determined by thresholded, logged gradient time-course.](image)
Figure 5. Example output for noisy ECG data, remedy: switch thresh.cardiac.initial_cpulse_select.method to 'auto'
If you want to use the method implemented in this toolbox, please describe it in your publication as follows:

“Correction for physiological noise was performed via RETROICOR [1,2] using Fourier expansions of different order for the estimated phases of cardiac pulsation (3rd order), respiration (4th order) and cardio-respiratory interactions (1st order) [2]: The corresponding confound regressors were created using the Matlab PhysIO Toolbox ([4], open source code available as part of the TAPAS software collection: http://www.translationalneuromodeling.org/tapas/).”

Our specific implementation of RETROICOR, uses Fourier expansions of different order for the estimated phases of cardiac pulsation (3\textsuperscript{rd} order), respiration (4\textsuperscript{th} order) and cardio-respiratory interactions (1\textsuperscript{st} order) following (Harvey et al., 2008).

**EXAMPLE DATASETS**

4.1 Philips

4.1.1 ECG3T

Courtesy of Sandra Iglesias, Translational Neuromodeling Unit, ETH & University of Zurich

4-electrode ECG and breathing belt, Philips 3T Achieva scanner

**Description:** Standard example; shows how to use scan counting either from beginning or end of run to synchronize physiological logfile with acquisition onsets of fMRI scans.
Figure 6: Influence of sqpar.Nprep. If Nprep is set (here = 3), the scan events including preparation gradients, dummies and scan volumes are counted from the start of the logfile (left), if Nprep is undefined, all is counted relative to the end of the logfile (right).
4.1.2 ECG7T

Courtesy of Zina-Mary Manjaly, University Hospital Zurich

4-electrode ECG and breathing belt, Philips 7T Achieva scanner

**Description:** The ECG data for ultra-high field data is typically much noisier than at 3 Tesla. Therefore, R-wave peaks are frequently missed by prospective trigger detection and not marked correctly in the logfile. This example shows how to select typical R-wave-peaks manually and let the algorithm find the heartbeat events.

![Central snippet of ECG-curve with chosen QRS-wave filter](image1)

![Correlation to chosen snippet - ECG smoothed with matched filter of 1 QRS-wave](image2)

![Raw ECG time course with detected heartbeat starts](image3)

Figure 7: Manual R-peak detection setting ECG$_{min}$ to 0.5. At 7T, this works more reliably than using the scanner logfile (blue stems), which misses some heartbeat events compared to the offline analysis of the script (green stems).
Figure 8: Output of Diagnostic raw time series (right) reveals that not all heartbeats have been detected when using a threshold of ECG_min=0.5 (left).

Figure 9: Output files. multiple_regressors contains the R-matrix for a GLM; test_phys.log is the modified SCANPHYSLOG.log now carrying all detected heartbeat and slice scan events and test_phys_ECG_kRpeakfile.mat may be used to rerun R-peak-detection identically
4.1.3 Pulse Oximeter 3T

Courtesy of Diana Wotruba, University and University Hospital of Zurich

PPU (finger plethysmograph) and breathing belt, Philips 3T Achieva scanner

**Description:** Similar to ECG3T, but a plethysmograph instead of an ECG was used to monitor the cardiac pulsation. Example shows how to extract heart and breathing rate.
4.1.4 ECG3T_Trigger

Courtesy of Tobias Hauser, Department of Child- and Adolescent Psychiatry, University of Zurich

Breathing belt, no ECG, Philips 3T Achieva scanner, patch (Roger Luechinger) to log scan event triggers into SCANPHYSLOG

**Description:** This logfile is very similar to the ECG3T-data above, but it doesn’t have an ECG attached. Interestingly, the scan events for every (2\textsuperscript{nd}? slice as initiated by the Philips scanner are logged in the SCANPHYSLOG-file enabling a direct evaluation of the toolbox’ algorithm to detect scan events from gradient time-course. A constant, small offset of 12 ms can be seen, which is constant over the whole session and thus absorbed in the RETROICOR cosine/sine phase expansion.

![Gradient X, Gradient Y, Gradient Z, scan events from logfile](image)

**Figure 10.** Lower panel: The scan events logged by the Philips system (purple) occur approximately 18 ms before the events detected by the toolbox algorithm, but only for every 2\textsuperscript{nd} slice. This constant offset, however, is absorbed by the phase expansion in cosine and sine regressors later on.

---

1 This example dataset is not included in the current release due to space limitations. Write kasper@biomed.ee.ethz.ch to retrieve a version.
4.2 GE

4.2.1 Pulse Oximeter 3T

Courtesy of Steffen Bollmann, Kinderspital Zurich and ETH Zurich

PPU (finger plethysmograph) and breathing belt, General Electric 3T scanner

**Description:** Similar to PPU, but acquired with on a GE system with two separate output logfiles for pulse oximetry and breathing amplitude, sampled with 40 Hz. The quality of the signal is particularly challenging, stemming from a patient population.
4.3 Siemens

4.3.1 ECG 3T

Courtesy of Miriam Sebold, Charite Berlin, and Quentin Huys, TNU Zurich

4-electrode ECG data, Siemens 3T scanner

**Description**: Similar to ECG 3T, but acquired on a Siemens system with only one logfile for ECG data. The quality of the signal is challenging, stemming from a patient population.
Physio is the main input structure to run the PhysIO Toolbox on a particular dataset. Its substructures (files, sqpar, thresh, model, verbose) are introduced in the following subsections and cover different parameter sets for different use cases of the toolbox. These substructures are altered in all examples/main_*.* files and should be as well in your own scripts, before you call physio_main_create_regressors. The code snippets documenting these substructures with its parameters are copied from the physio_new.m-file, where you will also find the latest version of the parameter documentation. Here is the header of that function:
function physio = physio_new(default_scheme, physio_in)
% creates complete PhysIO structure fed into physio_main_create_regressors
%   physio = physio_new(default_scheme, physio_in)
% IN
%   default_scheme - if set, default values for structure entries are set
%                     according to the application
%                     different templates are predefined, e.g.
%                     'empty' (default) - all strings are set to '', all
%                     numbers to []
%                     'RETROICOR' order of RETROICOR expansion taken
%                     from Harvey2008, JRMI28(6), p1337ff.
%                     'scan_timing_from_start'
%                     'manual_peak_select'
%   physio_in - used as input, only fields related to default_scheme
%                are overwritten, the others are kept as in physio_in
% OUT
%   physio - the complete physio structure, which can be used in
%            physio_main_create_regressors
% NOTE
%   All parameters used in the physIO toolbox are defined AND DOCUMENTED
%   in this file. Just scroll down and read through the comments!
% EXAMPLE
%   physio = physio_new('empty')
%   physio = physio_new('RETROICOR');
%   physio = physio_new('manual_peak_select', physio);
%   See also physio_main_create_regressors
% Author: Lars Kasper
% Created: 2013-04-23
% Copyright (C) 2013 TNU, Institute for Biomedical Engineering, University
% of Zurich and ETH Zurich.
% This file is part of the TNU CheckPhysRETROICOR toolbox, which is
% released under the terms of the GNU General Public
% Licence (GPL), version 3. You can redistribute it and/or modify it under
% the terms of the GPL
% (either version 3 or, at your option, any later version). For further
details, see the file
% COPYING or <http://www.gnu.org/licenses/>.
% $Id: physio_new.m 188 2013-05-05 14:06:25Z kasperla $
5.1 log_files

% structure containing general physiological log-file information

log_files.vendor = ''; % 'Philips', 'GE', or 'Siemens', depending on your
% MR Scanner system

log_files.cardiac = ''; % 'SCANPHYSLOG.log'; logfile with cardiac
data

log_files.respiration = ''; % 'SCANPHYSLOG.log'; logfile with
respiratory data

% (same as .cardiac
for Philips)

% log_files.sampling_interval = []; % in seconds, 2e-3 for Philips,
variable for GE,

% e.g. 40e-3
5.2 \texttt{sqpar}

\begin{verbatim}
\begin{verbatim}
sqpar.Nslices = [];  % number of slices per volume in fMRI scan
sqpar.NslicesPerBeat = [];  % usually equals Nslices, unless you
  trigger with the heart beat
sqpar.TR = [];  % volume repetition time in seconds
sqpar.Ndummies = [];  % number of dummy volumes
sqpar.Nscans = [];  % number of full volumes saved
(volumes in nifti file, preparation pulses
) sqpar.Nprep = [];  % set to >=0 to count scans and dummy
% number of non-dummy, volume like
 logfile is read from beginning, % before 1st dummy scan. If set,
last detected volume in the logfile % otherwise volumes are counted from
sqpar.TimeSliceToSlice = [];  % time between the acquisition of 2
subsequent % slices; typically TR/Nslices or
minTR/Nslices,
% if minimal temporal slice spacing
thresh.grad_direction % NOTE: only necessary, if
used
sqpar.onset_slice = 19;  % slice whose scan onset determines
% regressor timing to a particular
slice for the whole volume % volumes from beginning of run, i.e.
logfile, % includes counting of preparation
\end{verbatim}
\end{verbatim}
\end{verbatim}
### 5.3  thresh

#### 5.3.1  thresh.scan_timing

% determines thresholds used in preprocessing physiological logfiles,  
% either their timing (thresh.scan_timing) or the peripheral measures  
% itself (thresh.cardiac, thresh.respiration)  
thresh.scan_timing = []; % leave empty, if nominal scan timing,  
% derived from sqpar, shall be used

```plaintext
thresh.scan_timing.grad_direction = ''; % 'x', 'y', or 'z';  
% if set, sequence timing is calculated  
% from logged gradient timecourse along  
thresh.scan_timing.zero = []; % gradient values below this value are set to zero;  
% should be those which are unrelated to slice acquisition start  
thresh.scan_timing.slice = []; % minimum gradient amplitude to be exceeded when a slice scan starts  
thresh.scan_timing.vol = []; % minimum gradient amplitude to be exceeded when a new volume scan starts;  
% leave [], if volume events shall be determined as every Nslices-th scan event or via vol_spacing  
thresh.vol_spacing = []; % duration (in seconds) from last slice acq to % first slice of next volume;  
% leave [], if .vol-threshold shall be used
```
5.3.2 thresh.cardiac
thresh.cardiac = []; thresh.cardiac.modality = ''; % 'ECG','ECG_raw', or 'OXY' (for pulse oximetry), 'OXY_OLD', [deprecated]

% The initial cardiac pulse selection structure: Determines how the majority of cardiac pulses is detected
thresh.cardiac.initial_cpulse_select.method = 'load_from_logfile'; % 'load_from_logfile', 'manual' -(rather: threshold...autocorrelate?),
'load' thresh.cardiac.initial_cpulse_select.file = ''; % file containing reference ECG-peak (variable kRpeak)

% used for method 'manual' or 'load' [default: not set] string of file containing a
% if method == 'manual', this file is saved after picking the QRS-wave
% such that results are reproducible
thresh.cardiac.initial_cpulse_select.min = []; % threshold for correlation with QRS-wave to find cardiac pulses
thresh.cardiac.initial_cpulse_select.kRpeak = []; % variable saving an example cardiac QRS-wave to correlate with ECG time series

% The posthoc cardiac pulse selection structure: If only few (<20) cardiac pulses are missing in a session due to bad signal quality, a manual selection after visual inspection is possible using the following parameters. The results are saved for reproducibility.
thresh.cardiac.posthoc_cpulse_select.method = 'off'; % 'off', 'manual', 'load'
% 'off' - no manual selection of peaks
% 'manual' - pick and save additional peaks manually
% 'load' - load previously selected cardiac pulses
thresh.cardiac.posthoc_cpulse_select.file = ''; % filename where cardiac pulses are saved after manual picking

% Suspicious positions of missing or too many cardiac pulses are pre-selected by detecting outliers in histogram of heart-beat-2-heart-intervals
thresh.cardiac.posthoc_cpulse_select.percentile = 80; % percentile of beat-2-beat interval histogram that constitutes the "average heart beat duration" in the session
thresh.cardiac.posthoc_cpulse_select.upperThresh = 60; % minimum exceedance (in %) from average heartbeat duration to be classified as missing heartbeat
thresh.cardiac.posthoc_cpulse_select.lowerThresh = 60; % minimum reduction (in %) from average heartbeat duration to be classified an abundant heartbeat
5.4 model

% Determines the physiological noise model derived from preprocessed physiological data
   model.type = ''; % 'RETROICOR' - as in Glover et al, MRM 44, 2000
   model.input_other_multiple_regressors = ''; % other nuisance regressors to be included in design matrix
   model.output_multiple_regressors = ''; % output file for usage in SPM multiple_regressors GLM-specification

file with variable R
   model.input_other_multiple_regressors = ''; % other nuisance regressors to be included in design matrix
   model.output_multiple_regressors = ''; % output file for usage in SPM multiple_regressors GLM-specification

file with variable R
   model.order.c = []; % natural number, order of cardiac phase Fourier expansion
   model.order.r = []; % natural number, order of respiratory phase Fourier expansion
   model.order.cr = []; % natural number, order of cardiac-respiratory-phase-interaction Fourier expansion

28, 2008
   model.order.orthogonalise = 'none'; % string indicating which regressors shall be orthogonalised;
   model.order.orthogonalise = 'none'; % mainly needed, if acquisition was triggered to heartbeat (set to 'cardiac') OR
   model.order.orthogonalise = 'none'; % if session mean shall be evaluated (e.g. SNR-studies, set to 'all')
   model.order.orthogonalise = 'none'; % 'n' or 'none' - no orthogonalisation is performed
   model.order.orthogonalise = 'none'; % Possible Values
   model.order.orthogonalise = 'none'; % (default: 'none'
   model.order.orthogonalise = 'none'; % 'c' or 'cardiac' - only cardiac regressors are orthogonalised
   model.order.orthogonalise = 'none'; % 'r' or 'resp' - only respiration regressors are orthogonalised
   model.order.orthogonalise = 'none'; % 'mult' - only multiplicative regressors are orthogonalised
   model.order.orthogonalise = 'none'; % 'all' - all physiological regressors are orthogonalised to each other
5.5  verbose

% determines how many figures shall be generated to follow the workflow of the toolbox and whether the graphical output shall be saved (to a PostScript-file)
verbose.level = 1; % 0 = no graphical output; 1 = main plots (default);
% 2 = debugging plots, for setting up new study; 3 = all plots
verbose.fig_handles = {}; % collector of all generated figure handles during a run of physio_main_create_regressors
verbose.fig_output_file = ''; % file name (including extension) where to print all physIO output figures to,
% e.g. 'PhysIO_output.ps' or 'PhysIO_output.jpg'

% The specified extension determines how the figures will be saved
% .ps - all figures are saved to the same, multiple-page postscript-file
% .fig, .tiff, .jpg - one file is created for each figure, appended by its index, e.g.

'PhysIO_output_fig01.jpg'

7.2
Todo/Feature Requests

- compare end of file with last gradient slice logged in SCANPHYSLOG
- extra-systole modelling
- Manual: Use cases, Step-by-Step instructions/Inspections
  - use Matlab -> publish to make manual better readable and up to date
- merge these:
  - ons_secs.cpulse = physio_get_cardiac_pulses(ons_secs.t, ons_secs.c, ...)
    thresh.cardiac, verbose);

    [ons_secs, outliersHigh, outliersLow] = physio_correct_cardiac_pulses_manually(ons_secs,80,60,50);