BRAVEHEART: Open-Source Software for Automated Electrocardiographic and Vectorcardiographic Analysis

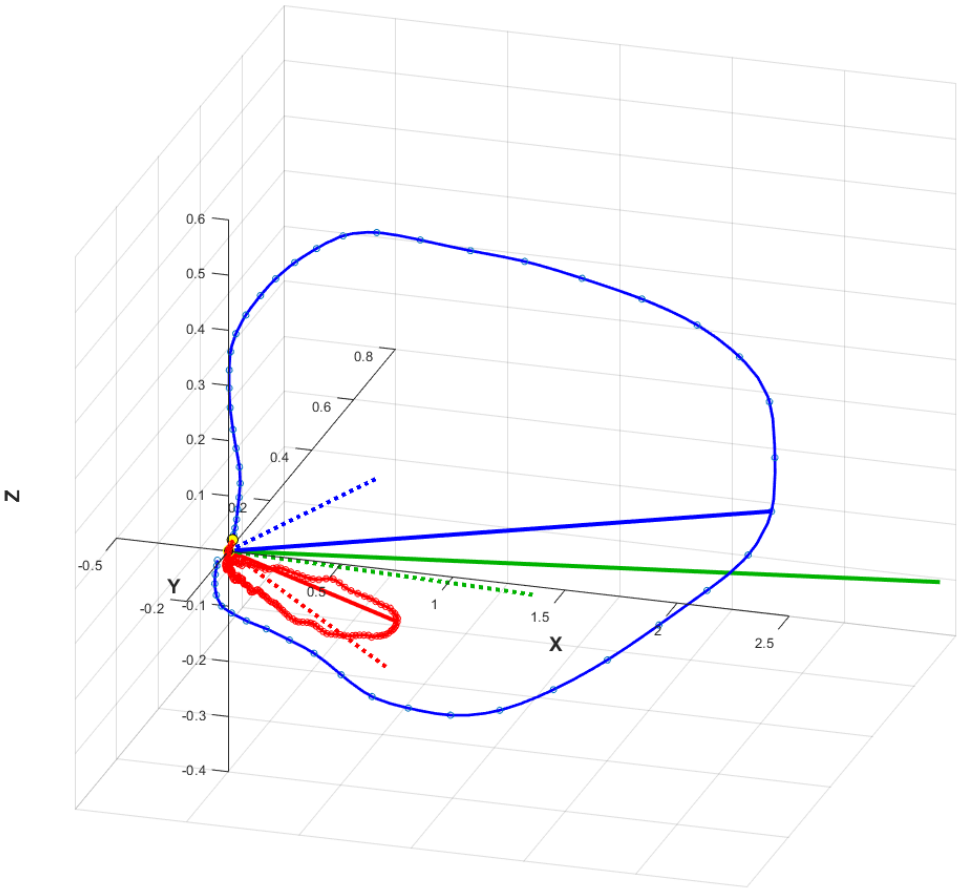
User Guide

Software Version 1.6.1

Hans Fredrich Stabenau, MD, PhD & Jonathan W. Waks, MD

Harvard-Thorndike Electrophysiology Institute, Department of Cardiovascular Medicine,
Beth Israel Deaconess Medical Center, Harvard Medical School, Boston, MA, USA

`http://www.github.com/BIVectors/BRAVEHEART`



Document Version 9/22/2025

Contents

1 Introduction

BRAVEHEART (**B**eth Is**r**ael **A**nalysis of **Ve**ctors of the **Heart**) is a modular, customizable, open-source software package for processing electrocardiograms (ECGs) and vectorcardiograms (VCGs) for research purposes. BRAVEHEART was built using MATLAB and requires a version after R2022a (`http://www.mathworks.com`). The most up to date version of the software can be found on GitHub at `http://www.github.com/BIVectors/BRAVEHEART`, where the software, source code, and executables for Windows and Mac are available under version 3 of the General Public License (GPL) (`http://www.gnu.org/licenses/gpl-3.0.en.html`).

The following document contains information on how to use BRAVEHEART and how to efficiently edit select parts of the source code to allow addition of new ECG formats, transformation matrices, ECG/VCG parameters, and new denoising/filtering methods. Information on the inner-workings of the software is also provided to facilitate understanding exactly what the software is doing at each step of ECG/VCG processing. This document is focused on use of the BRAVEHEART software. For additional, detailed information on the methodology behind various aspects of signal processing and annotation, please see our separate methods manuscript [1] which is also available on the BRAVEHEART GitHub repository as file `braveheart_methods.pdf`.

BRAVEHEART is available in different packages that will fit the needs of different types of users. `braveheart_batch.m` is a command line based program without any graphical user interface (GUI) that is able to rapidly process directories of ECGs by setting a few parameters. `braveheart_gui.m` is a GUI based version of BRAVEHEART that allows additional control over every aspect of ECG/VCG processing and additional types of signal processing and data visualizations. Both the command line version and GUI version of BRAVEHEART are also available as compiled executables for Windows and Mac computers that can be run without the need to have MATLAB installed.

We hope that BRAVEHEART will be helpful for your ECG/VCG research projects and welcome any feedback, bug reports, criticisms, or suggestions that you have while using BRAVEHEART. We can be contacted at `braveheart.ecg@gmail.com` with any questions, suggestions, or concerns.

2 Version History/Changelog

- **v1.0.0** – 5/1/2023: Initial public release
- **v1.0.1** – 9/12/2023: See GitHub Releases for detailed change log.
  - Added additional mother wavelets/wavelet levels to the GUI dropdowns.
  - Added error handling for selecting wavelet decomposition levels > max level.
  - Default high-frequency denoising (low-pass filtering) wavelet decomposition level in `Annoparams.m` and `Annoparams.csv` set to 1 (was 2). This changes the low-pass cutoff from 62.5 Hz to 125 Hz for an ECG sampled at 500 Hz. The default value can be changed by editing `Annoparams.m` when running via MATLAB or `Annoparams.csv` when running via executable.
  - When choosing a new ECG format in the GUI, the default low-frequency denoising (high-pass filtering) is set to the highest level with frequency cutoff < 0.25 Hz (see **Chapter 7**).
  - Updated normal range values based on our new publication [2] (see **Chapter 22**).
  - Added ability to read Cardiosoft XML files (see **Chapter 23**).
- **v1.0.2** – 9/29/2023: See GitHub Releases for detailed change log.
  - Fixed a bug related to median beat annotations being incorrect by a small number of samples if the ECG sampling frequency was not 500 Hz or 1000 Hz.
  - Became aware of a bug that results in parallel ECG batch processing not working only when using a compiled executable version of BRAVEHEART (both GUI and command line). While a fix is being worked on, parallel processing via the compiled versions of BRAVEHEART has been disabled.
  - Fixed a bug that resulted in errors which prevented calculating results in the `LeadMorphology.m` results class for ECGs sampled at < 500 Hz.
  - Updated functions to load DICOM, HL7 XML, and Cardiosoft XML ECG files to deal with different versions of these file formats (see **Chapter 23**).
  - Fixed issue that caused the Offsets and **Lead Morphology** button/functions to not display properly if an ECG lead was missing (see **Chapter 12.2** and **Chapter 12.6**).
- **v1.1.0** – 10/5/2023: See GitHub Releases for detailed change log.
  - Refactored quality assessment to fix the bug that caused an error when using parallel ECG batch processing in a compiled executable version of BRAVEHEART. The external file `quality_presets.csv` was renamed `Qualparams.csv`. While running BRAVEHEART via MATLAB source code, quality parameters are now set using the file `Qualparams.m` which is analagous to `Annoparams.m`. The external file `Qualparams.csv` is only used when running BRAVEHEART via compiled executables. See **Chapter 16.3** for further details of these changes.
  - Fixed bug that caused errors when running parallel batch processing via compiled executables.
  - Added quality test cases to `test_braveheart.m` (see **Chapter 30**).
  - Added console logging when using executable versions to assist with troubleshooting any errors (see **Chapter 31.7**).
- **v1.1.1** – 12/6/2023: See GitHub Releases for detailed change log.
  - Added function to calculate the QRS frontal plane axis (see **Chapter 27**).
  - Added QRS area, T area, and QRST area for the 12 standard ECG median beats to `Lead_Morphology.m` results class.
  - Added new figure that graphically displays QRS fontal plane axis when clicking on the

Lead Morphology button (see **Chapter 12.2** and **Chapter 12.6**).

  - Added quality test cases to `test_braveheart.m` reflecting additions to `Lead_Morphology.m` (see **Chapter 30**).
  - Updated angle information figure displayed by clicking on the Angle Info button (See **Chapter 12.7**) to better clarify the definition of angle elevation in spherical coordinates.
  - Fixed some bugs related to processing ECGs with sampling frequencies other than 500 Hz.

- **v1.1.2** – 2/27/2024: See GitHub Releases for detailed change log.
  - Updated `load_philipsxml.m` to allow loading of more variations in Philips XML format files.
  - Fixed bug that prevented display of the Filtering figure in the GUI (see **Chapter 20.13**).
  - Mac OS: Fixed bug that prevented display of external reference files (userguide, equations, etc) from their respective buttons when running the compiled GUI.
  - Added some missing file format extensions to `get_source_ext.m`.

- **v1.1.3** – 3/22/2024: See GitHub Releases for detailed change log.
  - Modified `load_dicom.m`, to be more robust to variations in labeling of leads within DICOM (.dcm) files.
  - Added the ability to read most SCP-ECG format (.scp) ECGs utilizing open source code from The BioSig Project v3.8.4 (`https://biosig.sourceforge.net`) (see **Chapter 23**).
  - Added the ability to read European Data Format (EDF) (.edf) ECGs. See **Chapter 23** for some important information regarding how BRAVEHEART deals with these files.
  - Added the ability to read Physionet format .csv files (see **Chapter 23**).
  - Added the ability to read ASCII text files exported from the Abbott WorkMate Claris EP recording system. Loading ECGs in this format is complex - please see **Chapter 23** for additional information.
  - Fixed bug that prevented ECG Splitter from working properly (see **Chapter 12.7** for additional information).

- **v1.2.0** – 5/15/2024: See GitHub Releases for detailed change log.
  - Modified `load_edf.m`, to be more robust to variations in labeling of leads within EDF (.edf) files.
  - Added speed-time integrals for the QRST complex, QRS complex, and T wave to `VCG_Calc.m` results class (see **Chapter 24.3**).
  - Added the area under the VM QRS complex and VM T wave to `VCG_Calc.m` results class (see **Chapter 24.3**).
  - Rewrote code to calculate max/min speed in QRST complex, QRS complex, and T wave to improve efficiency and readability.
  - Renamed annotation parameter `blanking_samples`, which is the value for the blanking window after QRS onset used for speed calculations in `Annoparams.m`, to `blanking_window_q`.
  - Added annotation parameter `blanking_window_t`, which is the value for the blanking window after T wave onset used for speed calculations, to `Annoparams.m`. Previously this was always set to 0, resulting in the possibility of incorrect max speed or max speed location in the T wave due if the location of QRS offset was too early by a few samples. The nominal value is `blanking_window_t = 20`, which ignores the first 20 ms of the T wave when calculating max/min/median speed.
  - Speed blanking windows, `blanking_widndow_q` and `blanking_window_t`, now take values in milliseconds instead of samples to allow more standardized blanking when using ECGs with different sampling frequencies. (see **Chapter 6**).

- Edited and reorganized GUI Section 6.5 which displays VCG speed information: Added a text box so both `blanking_window_q` and `blanking_window_t` can be edited in the GUI. The speed graph X-axis units are now now milliseconds from Q_{on}(so that the Q_{on}point is time 0 ms). See **Chapter 12.7 Section 6.5** for further details.

- **v1.2.1** – 6/20/2024: See GitHub Releases for detailed change log.
  - Rewrote code to calculate normal ranges for various VCG parameters to improve efficiency and readability. See **Chapter 22** for further details.
  - BRAVEHEART GUI now extracts information on race from XML/DICOM files for the purposes of calculating normal value ranges. If no information on race is available BRAVEHEART assumes that the patient is white. The default race can be changed by editing line 3184 of `braveheart_gui.m`. BRAVEHEART Batch does not extract demographic information as this is only used for displaying normal ranges in the GUI.
  - BRAVEHEART GUI now extracts information on age, sex, and race from DICOM files and HL7 XML files in addition to all other supported XML formats.
  - GUI features for changing demographics have been changed. BRAVEHEART no longer assumes anything about missing values of age, race, sex, and BMI. If any of these values are missing, the appropriate normal ranges are calculated using margins. See user guide **Chapter 22** and **Chapter 12.7** for further details.
  - Added buttons to GUI section 6.6 (see **Chapter 12.7**) to open the ECG file and to display select demographic data from XML files.
  - If the load directory option is chosen when loading a directory, a counter is now displayed with the specific ECG number and the total number of ECGs that can be loaded using the next and previous ECG buttons. See **Chapter 12.2** for further details.
  - Added new variable `VM_max_rpk_loc` which returns the time (in ms) after QRS onset of the maximum R peak in lead VM. This corresponds to the time of maximal distance of the VCG loop from the origin. See **Chapter 24.1** for further details.
  - Added keyboard shortcuts for common actions within the BRAVEHEART GUI. See **Chapter 26** for further details.
  - Fixed a bug that resulted in incorrect localization of the T wave maximum voltage value in there was very large ST segement deviation.
  - Added ability to read Schiller XML format. See **Chapter 23** for further details.
  - Installation of the Mac executable has been improved and significantly simplified. Installation no longer requires the `.command` script to copy .csv files to `~/home/BRAVHEART`. The .csv files now reside in the directory where the executable is stored and they can be edited as needed from this location rather than `~/home/BRAVHEART`. Installation instructions in **Chapter 4** have been updated.

- **v1.2.2** – 8/30/2024: See GitHub Releases for detailed change log.
  - Added quality probability to the Summary Figure (see **Figure 50**).
  - Added ability to directly read Physionet .dat files and their associated header files (.hea). See **Chapter 23** for further details.
  - Added some basic heart rate variability measures. See **Chapter 24.4** for further details.
  - Added ECG path and filename, individual beat statistics (`Beat_Stats`) results, quality metrics, and annotation parameters to exported .mat files. See **Chapter 25** for further details.
  - The location of R peaks for beats that are initially detected but then removed via PVC detection, outlier detection, manually, or due to incomplete QRST complexes usually during the very start/end of an ECG recording, are now stored in the `Beats` results class. See **Chapter 10.1.1** for further details.

  - Frequency, units per mV, and data orientation (columns or rows) can be adjusted for the "Generic .csv" ECG file format using the external file `generic_csv_params.csv`. This allows users to adjust these attributes without needing to edit code in any MATLAB files, and will allow users running BRAVEHEART via compiled executable to edit these variables as needed for their specific ECG data. See **Chapter 23** for additional details.
  - Added T wave mechanical dispersion (TMD) and T wave residuum (TWR) to the `VCG_Morphology` results class (see **Chapter 27** for further details).

- **v1.3.0** – 10/13/2024: See GitHub Releases for detailed change log.
  - Added ability to filter out the T wave during R peak detection. This may be useful in certain situations where there are very large T waves. Further details can be found in **Chapter 15.1**.
  - Added `pkfilter` to `Annoparams.m` to turn on/off R peak filtering (see **Chapter 6** and **Chapter 15.1**).
  - Updated the Threshold figure to show the filtered VM lead used for R peak detection when filtering is being used for R peak detection (see **Chapter 15.1** and **Figure 25**).
  - Fixed a bug where certain SCP-ECG (.scp) format ECGs would throw an error if there were issues reading non-ECG signal patient data (name, date of birth, etc.) encoded within the files.

- **v1.3.1** – 1/6/2025: See GitHub Releases for detailed change log.
  - Added ability to read Megacare XML ECG files (see **Chapter 23**).
  - Added ability to read Norav 1200M raw data type (.rdt) ECG files (see **Chapter 23**).
  - Added parameters to `generic_csv_params.csv` that allow the user to control the starting row/column to account for headers or other offsets, and the ability to specify the lead order. This will allow users to use a wider variety of .csv files without having to adjust formatting within the files. See **Chapter 23** for additional details.

- **v1.4.0** – 3/6/2025: See GitHub Releases for detailed change log.
  - Added customizable light/dark mode colors to GUI. Dark mode and light mode can be toggled in the Utilities section of the GUI. See **Chapter 28** for further details.
  - Fixed a bug and rewrote code for T wave maximum detection. Previously, if a lead's T wave amplitude was so small that BRAVEHEART could not find a true T wave max, it would assign the lead's T wave maximum location as the location of the T wave maximum in the VM lead as an approximation. Now, T wave maximum detection is more robust and uses the MATLAB function `findpeaks` instead of looking for raw maximum values. If a T wave maximum is not found that is at least 0.01 mV, BRAVEHEART now reports the T wave maximum and location as `NaN` to be more accurate. ****NOTE**** that as a result of this change, some very low T wave maximum amplitudes and their corresponding locations will now report as `NaN` or may change from prior versions. In general, this change should have no effect on normal sized T wave deflections and does not affect the location of the T wave max in the VM lead.
  - Fixed a bug where the parameter `pkfilter` was missing from `Annoparams.csv` and therefore could not be set when running BRAVEHEART via a compiled executable. In practice, this would set `pkthresh = 0`, but it only affected running the compiled command line version of BRAVEHEART, as the value for `pkthresh` could be set via checkbox in the GUI (see **Chapter 12.2**).
  - The Lead Morphology figure in the GUI (see **Figure 58**) now allows re-scaling of Y-axes. See **Chapter 20** for further details.
  - Added save buttons to figures. This replaces the save checkbox in the GUI. See **Chapter 12.8** for further details.
  - Reorganized GUI buttons to shift fiducial points. See **Chapter 12.9** for further details.

  - Reorderd lead avR, avL, and avF when displaying the 12-lead ECG figure (**Figure 52**) to be consistent with the typical display order.
  - Increased the default value of the low frequency noise detection threshold `lf_noise` in `Qualparams.m` or `Qualparams.csv` from 20 uV to 30 uV. See **Chapter 16.3** for further details.
  - Improved error handling for the Lead Morphology results class so that unexpected errors do not result in values of `NaN` for all measurements.
  - Added ability to read .mat files exported from BRAVEHEART as ECG data. See **Chapter 23** for further details.
  - Added ability to read binary .dat files from Edan SE 601C ECG machines. See **Chapter 23** for further details.
  - Fixed a bug where displaying VCG acceleration in the VCG Speed tab of the GUI (**Chapter 12.7**) would cause a crash.
  - Fixed a bug where very large pacing spikes that were greater in amplitude than QRS amplitude were incorrectly being measured as the R and S wave amplitudes.
  - Fixed a bug where the location of VM T max did not change if fiducial point locations were manually edited.
  - Fixed other minor visual bugs in the GUI.

- **v1.5.0** – 7/1/2025: See GitHub Releases for detailed change log.
  - Added new method utilizing continuous wavelet transform (CWT) for pacemaker spike detection and removal. See **Chapter 15.2** for further details.
  - Updated GUI to allow editing of parameters that control CWT method of pacemaker spike detection and removal.
  - Added new data to exported .mat files related to pacemaker spike detection and removal. See **Chapter 25** for further details.

- **v1.5.1** – 7/17/2025: See GitHub Releases for detailed change log.
  - Fixed a bug where ECGs without pacing would sometimes be erroneously labeled as paced (`pacing_detected = 1` in output files) using the median filter method of pacemaker spike detection.

- **v1.6.0** – 8/23/2025: See GitHub Releases for detailed change log.
  - Added ability to read MFER (.mwf) ECG files. See **Chapter 23** for further details.
  - Added ability to convert an entire directory of ECG files into `unformatted` format. See **Chapter 12.7** for further details.
  - Changed the format specifier from `%f` to `%.15g` when writing unformatted ECG files to remove trailing zeros and save disk space.
  - Adjusted GUI so that it displays correctly in MATLAB R2025a and future MATLAB releases. The graphics engine for MATLAB has been redone in R2025a, and as a result of this and phasing out of the GUIDE GUI editor, the BRAVEHEART GUI was not displaying as expected when using R2025a. When loading the BRAVEHEART GUI in R2025a it will look somewhat different than when loading in versions prior to R2025a, but the functionality is the same. Users can always use a version prior to R2025a if they prefer.

- **v1.6.1** – 9/22/2025: See GitHub Releases for detailed change log.
  - Fixed a bug where `pacing_detected` in output files was incorrectly being set to `1` when no pacing had been detected. When using the CWT spike filter these incorrect values of `pacing_detected = 1` would have a blank value for `num_paced_leads`.

- Fixed issues with text scaling in various figures when using R2025 on a Mac.
- Added new function `xyz2azel.m` which converts Cartesian coordinates into spherical coordinates with BRAVEHEART specific angle conventions (+X towards left, +Y down, +Z posterior).
- Updated user guide with additional information on conventions for azimuth and elevation and transformation between spherical and Cartesian coordinates (see **Figure 71**).
- Enforced orientation of the new basis vectors (right singular vectors, **V**) that are used to describe the best fit QRS or T planes (`qrs_loop_normal` and `t_loop_normal` , see **Chapter 24.2** and **Chapter 27**) so that the Z-component of the 3rd right singular vector (normal to the best fit plane) is always negative (anterior), the X-component of the 1st right singular vector is always positive (leftward), and the Y-component of the 2nd right singular vector follows a right-handed convention. This has no effect on any measurements other than the signs of the components of the normal vectors to the best fit plane. (`qrs_loop_normal` and `t_loop_normal`) for which the only change may be in the sign of one or more vector components. This was done to allow consistency between different versions of MATLAB as the signs of the singular vectors are not uniquely defined after singular value decomposition.
- Added new variables `qrs_loop_plane_az` , `qrs_loop_plane_el` , `t_loop_plane_az` , and `t_loop_plane_az` which report the orientation of the normal vector to the best fit QRS or T plane as azimuth and elevation. Since the normal vectors are unit vectors with magnitude 1, the magnitude of the normal vector is not reported.
- Added new variables `xy_qrs_loop_dir` , `xy_qrs_signed_area` , `xz_qrs_loop_dir` , `xz_qrs_signed_area` , `zy_qrs_loop_dir` , `zy_qrs_signed_area` , `best_qrs_loop_dir` , and `best_qrs_signed_area` which report the direction of QRS loop rotation (clockwise [CW], counterclockwise [CCW], or Indeterminate) in the XY (frontal), XZ (transverse looking up from feet with back posterior), ZY (left sagital), and the best fit QRS plane viewed from the direction of the normal vector obtained with singular value decomposition, based on signed area and specific viewing angle and axes orientation. See userguide **Chapter 24.2** and **Chapter 27** for further details.
- Updated the VCG Loops figure in the GUI (**Figure 56**) to show the direction of QRS loop propagation. This can be toggled off if desired.

3 Abbreviations

- BIDMC – Beth Israel Deaconess Medical Center
- BMI – body mass index
- bpm – beats per minute
- CWT – continuous wavelet transform
- DC – direct current
- DICOM – Digital Imaging and Communications in Medicine
- ECG – electrocardiogram
- GPL – general public license
- GUI – graphical user interface
- HL7 – Health Level Seven
- HR – heart rate
- HRV – heart rate variability
- ISHNE – International Society for Holter and Noninvasive Electrocardiology
- LSTM – long-short term memory
- LVH – left ventricular hypertrophy
- NN – neural network
- NCC – normalized cross correlation
- OS – operating system
- PVC – premature ventricular contraction
- Q_{on} – QRS complex onset
- Q_{off} – QRS complex end/offset
- RMSE – root mean square error
- SNR – signal to noise ratio
- SVD – singular value decomposition
- SVG – spatial ventricular gradient
- TMD – T wave mechanical dispersion
- T_{off} – T wave end/offset
- TWR – T wave residuum
- VCG – vectorcardiogram
- VM – vector magnitude
- XML – extensible markup language

4 Installation

The most up to date MATLAB source code and executables for Windows and Mac operating systems can be found on the BRAVEHEART GitHub repository at `http://www.github.com/BIVectors/BRAVEHEART`.

4.1 Using MATLAB Source Code (For Users With MATLAB)

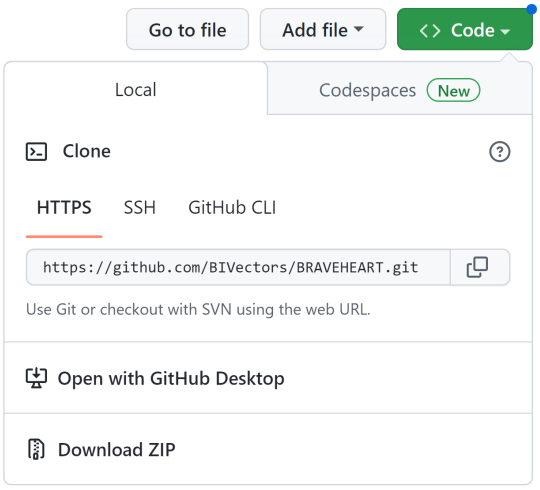
Using BRAVEHEART via source code **requires MATLAB R2022a or later** and the following toolboxes (`https://www.mathworks.com/products.html`):

- Signal Processing
- Deep Learning
- Wavelet
- Parallel Computing (optional)
- Image Processing (optional – only for reading DICOM files – see **Chapter 23**)

We have tested BRAVEHEART on versions of MATLAB up to 2024b.

To get a copy of the latest source code, open the BRAVEHEART GitHub and click on the 

button:



From here you can download a `.zip` file of all of the source code or clone the GitHub repository to your computer.

Alternatively, you can click the 'Releases' link on the right hand side, or go directly to

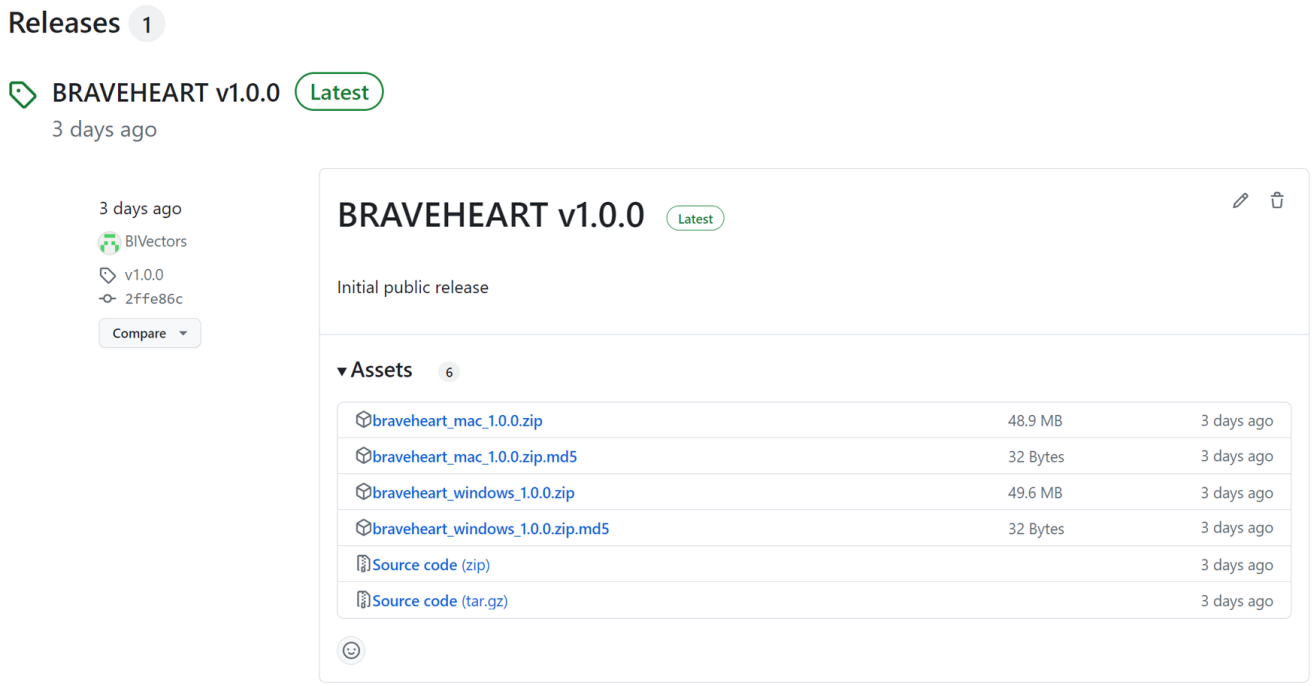
`http://github.com/BIVectors/BRAVEHEART/releases` and download a `.zip` file of all of the source code for a specific release. If you download the source code this way make sure you obtain the most recent release.

You can now view/edit/run the source code:

- To run the GUI version of BRAVEHEART run `braveheart_gui` from within MATLAB as described in detail in **Chapter 12**.
- To run the command line version of BRAVEHEART run `braveheart_batch()` from within MATLAB as described in detail in **Chapter 13**.

4.2 Using Executables (For Users Without MATLAB)

If you do not have access to MATLAB, or if you have MATLAB but do not have access to all of the required toolboxes, BRAVEHEART can be run via a compiled executable which contains the core MATLAB functions and all of the required and optional MATLAB toolboxes. Open the BRAVEHEART GitHub and navigate to the 'Releases' link on the right hand side, or go directly to `http://github.com/BIVectors/BRAVEHEART/releases`.



Note that the most current release will be after v1.0.0 and we suggest that you download the

most recent version of the software. Download the `.zip` file that is appropriate for your operating system, unzip it, and then follow the instructions below. The files `braveheart_windows_x.x.x.zip.md5` and `braveheart_mac_x.x.x.zip.md5` contain the MD5 hashes for their respective `.zip` files (where `x.x.x` is the release version) if you would like to verify the contents of the `.zip` file before unzipping.

You must unzip/install the executable in a folder that has write access so that various files that are required for BRAVEHEART to function are editable.

4.2.1 Install the MATLAB Runtime

The MATLAB Runtime is a set of libraries that allow execution of compiled MATLAB code. The **R2022a (9.12)** version of the Runtime must be installed to run any version of the compiled BRAVEHEART programs.

The MATLAB Runtime only needs to be installed once. To install there are 2 options:

1. Go to `https://www.mathworks.com/products/compiler/matlab-runtime.html` and download the **R2022a (9.12)** Runtime for Windows or Mac.
2. Within the downloaded `.zip` file there is a shortcut to directly download the appropriate (Windows or Mac) Runtime version.

After downloading, run the MATLAB Runtime installer and follow the instructions to complete the installation. Depending on the security settings of your computer you may need administrator rights to install the MATLAB Runtime.

IMPORTANT: If you manually download the MATLAB runtime from the Mathworks website, make sure you download the **R2022a (9.12)** version of the Runtime.

4.2.2 Install BRAVHEART Executables on a Windows PC

After installing MATLAB Runtime R2022a (see above), unzip the `braveheart_windows_x.x.x.zip` file (where x.x.x is the version number) in a directory that has write access (on some systems, depending on security settings, the 'Program Files' folder may not work). You will see the following 13 files/folders:

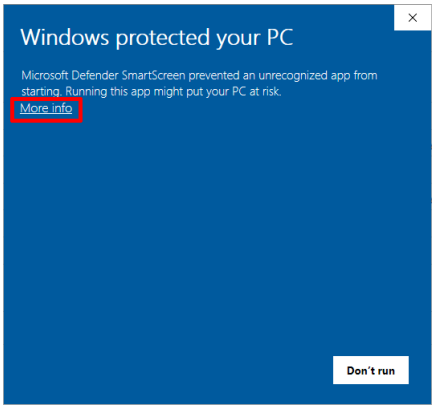
- `braveheart_gui.exe` – BRAVEHEART GUI version (see **Chapter 12**).
- `braveheart_batch.exe` – BRAVEHEART Command line version (**Chapter 13**).
- `Annoparams.csv` – File to edit annotation parameters (**Chapter 6**).
- `ecg_formats.csv` – File to add new ECG formats (**Chapter 8**).
- `transform_mats.csv` – File to add new transformation matrices (**Chapter 9**).
- `batch_settings.csv` – File to edit command line batch parameters (**Chapter 13**).
- `Qualparams.csv` – File to edit quality parameters (**Chapter 16**).
- `search_presents.csv` – File to edit annotation presets (**Chapter 21**).
- `generic_csv_params.csv` – File to edit how "generic .csv" load (**Chapter 23**).
- `braveheart_userguide.pdf` – This file.
- `Example ECGs` – Directory containing 4 example ECG files (**Chapter 5**).
- `Download MATLAB Runtime - Win 64 bit` – Link to download MATLAB Runtime.
- `splash.png` – Image used while loading.

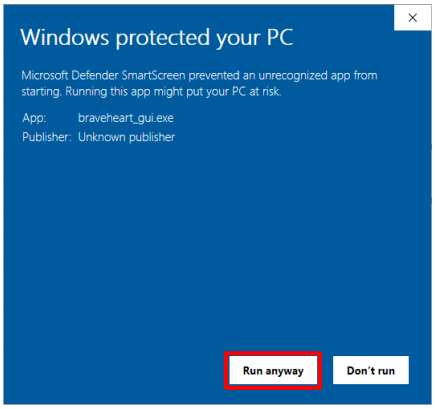
The external `.csv` files can be edited as needed and described in later sections of the user guide. If you are unable to edit the `.csv` files, unzip `braveheart_windows_x.x.x.zip` in a different directory with write access. The `.csv` files need to be in the same directory as the `.exe` files.

Double click on `braveheart_batch.exe` or `braveheart_gui.exe` to run the program.

Whitelist BRAVEHEART on Windows Runtime Protection

Depending on your version of windows and your anti-virus software, when you run `braveheart_gui.exe` or `braveheart_batch.exe` for the first time, a pop-up may block execution of the program. If the Windows Defender pop-up blocks execution of the program simply click "**More Info**" and then "**Run anyway**":





The exact appearance and text may vary depending on your version of Windows and specific security settings. If an alternative security program is in use, there will be a similar way of whitelisting the executable so that in the future, when you run it, the OS will not block it. This process will only have to be repeated if you install a new version of BRAVEHEART.

4.2.3 Install BRAVHEART Executables on a Mac

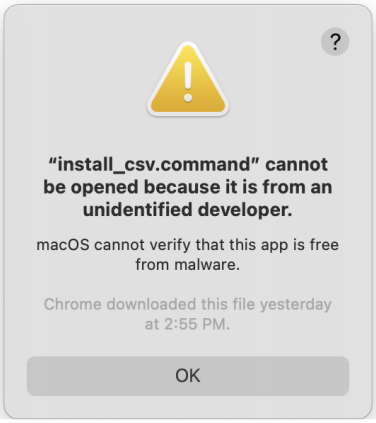
IMPORTANT: As of version 1.2.1, the way the Mac executable handles external .csv files has changed significantly (for the better). As as result, the installation steps outlined here will NOT work for versions prior to 1.2.1. If you are using a version prior to 1.2.1 please refer to the installation guide within the version of the user guide supplied with the executable.

After installing MATLAB Runtime R2022a (see above), unzip the `braveheart_mac_x.x.x.zip` file (where x.x.x is the version number) in a directory that has write access. You will see the following 16 files/folders:

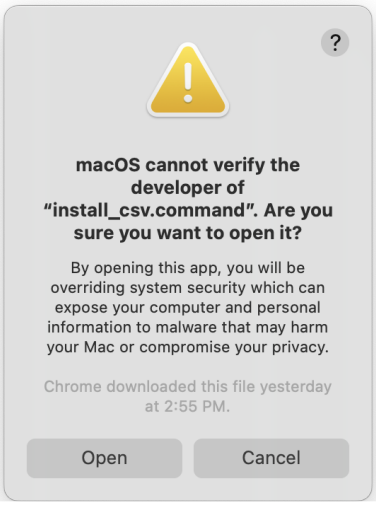
- `install.command` – Script to disable Mac OS runtime protection for relevant files (see next paragraphs in this section).
- `braveheart_gui.app` – BRAVEHEART GUI version (**Chapter 12**)
- `braveheart_batch.app` – BRAVEHEART Command line version (**Chapter 13**).
- `run_braveheart_gui.sh` – Script used by MATLAB
- `run_braveheart_batch.sh` – Script used by MATLAB
- `Annoparams.csv` – File to edit annotation parameters (**Chapter 6**).
- `ecg_formats.csv` – File to add new ECG formats (**Chapter 8**).
- `transform_mats.csv` – File to add new transformation matrices (**Chapter 9**).

- `batch_settings.csv` – File to edit command line batch parameters (**Chapter 13**).
- `Qualparams.csv` – File to edit quality parameters (**Chapter 16**).
- `search_presents.csv` – File to edit annotation presets (**Chapter 21**).
- `generic_csv_params.csv` – File to edit how "generic .csv" files load (**Chapter 23**).
- `braveheart_userguide.pdf` – This file.
- `Example ECGs` – Directory containing 4 example ECG files (**Chapter 5**).
- `Download MATLAB Runtime - Mac 64 bit` – Link to download MATLAB Runtime.
- `splash.png` – Image used while loading.

Double click `install.command`. The first time that you run `install.command`, Mac OS runtime protection may block it because it is from an "unidentified developer" and not signed:



To allow the script to run (this only has to be done once), **Control-click** `install.command`, and choose "**Open**" from the menu. When the popup below appears, click Open:



This process will only have to be repeated if you install a new version of BRAVEHEART.

A script will run in Terminal. Enter your system password and press **return** on the keyboard.

Entering your system password is necessary because the script whitelists `braveheart_gui.app`, `run_braveheart_gui.sh`, `braveheart_batch.app`, and `run_braveheart_batch.sh` for the Mac OS Gatekeeper runtime protection so the software can run without needing to manually whitelist each file. This is performed by removing the file attributes that blacklist an unsigned file from Gatekeeper and requires your system password:

```
sudo xattr -d com.apple.quarantine <filename>
```

If you prefer to not run the script, you will have to manually whitelist the `.app` and `.sh` files by **Control-clicking** on each file and choosing "**Open**" from the menu.

Double click on `braveheart_batch.app` or `braveheart_gui.app` to run the program. The `.csv` files need to be in the same directory as the `.app` files.

5 Quick Start Using the BRAVEHEART GUI

The following sections are provided as a way to "quick start" use of the BRAVEHEART GUI. Details on using BRAVEHEART in command line mode are provided in **Chapter 13**. Your understanding of the BRAVEHEART GUI will be better if you understand `Annoparams.m` (**Chapter 6**) and the specifics functions in each GUI section (**Chapter 12**), but this section is designed to get you using BRAVEHEART quickly, while showing you commonly used functions.

This quick start guide uses 4 ECG files, (`example1.xml`, `example2.xml`, `example3.xml`, and `example4.xml`) which are available on the BRAVEHEART GitHub repository in GE MUSE XML format (see **Chapter 23**).

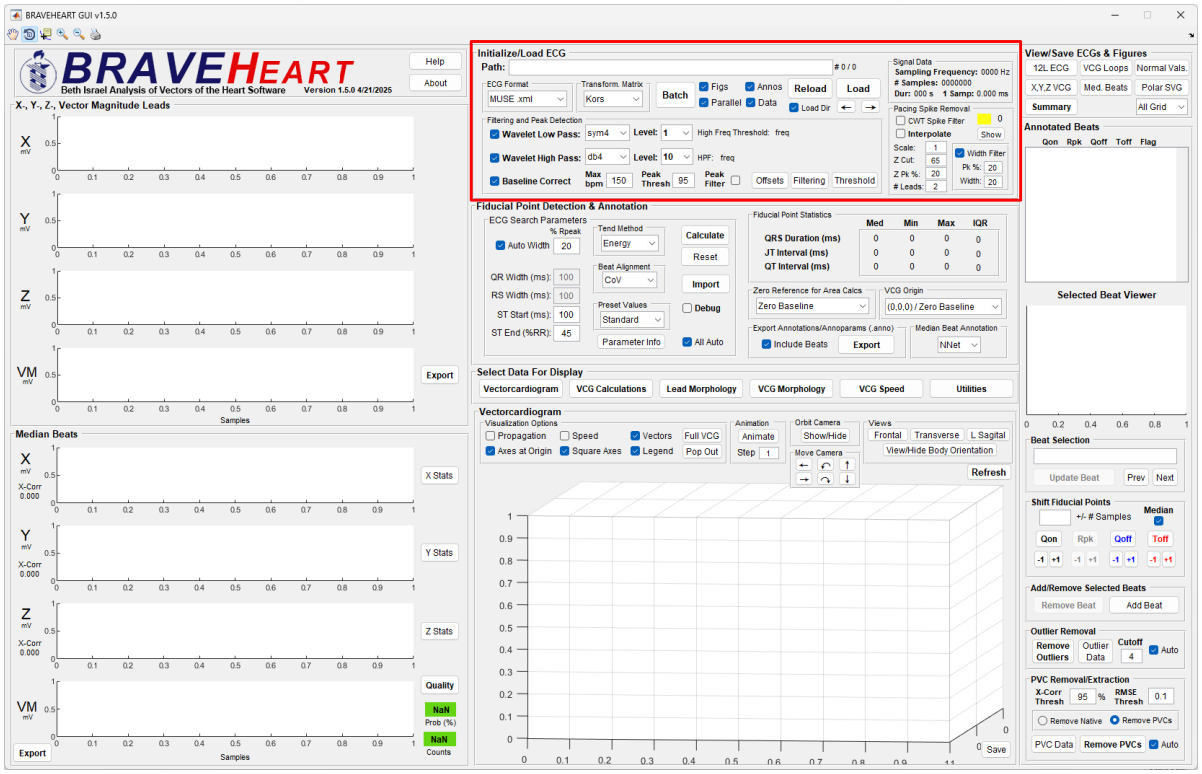
IMPORTANT: For the GUI to display properly your screen resolution must be at least 1920 x 1080. If your screen resolution is set appropriately but the GUI still does not display properly, your computer display settings likely have some form of scaling turned on; this setting increases the size of text to improve readability, but also effectively reduces the screen resolution. You can disable resolution scaling as shown in **Chapter 31.1**.

IMPORTANT: If you are using MATLAB R2025a or later, please make sure that you are using **BRAVEHEART version 1.6.0 or later**. Due to a graphics overhaul in R2025a, the GUI in BRAVHEART versions prior to 1.6.0 will not display correctly.

5.1 Example 1 - Good Quality ECG

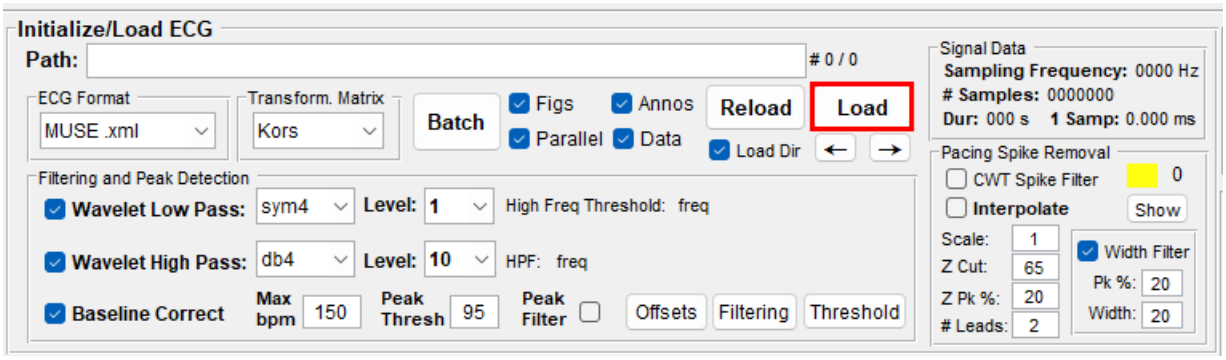
In this example we will load and process an ECG without any ectopy or significant noise/artifact using the standard filtering and annotation settings, and save the results to an external file.

Start at the highlighted '**Initialize/Load ECG**' section:



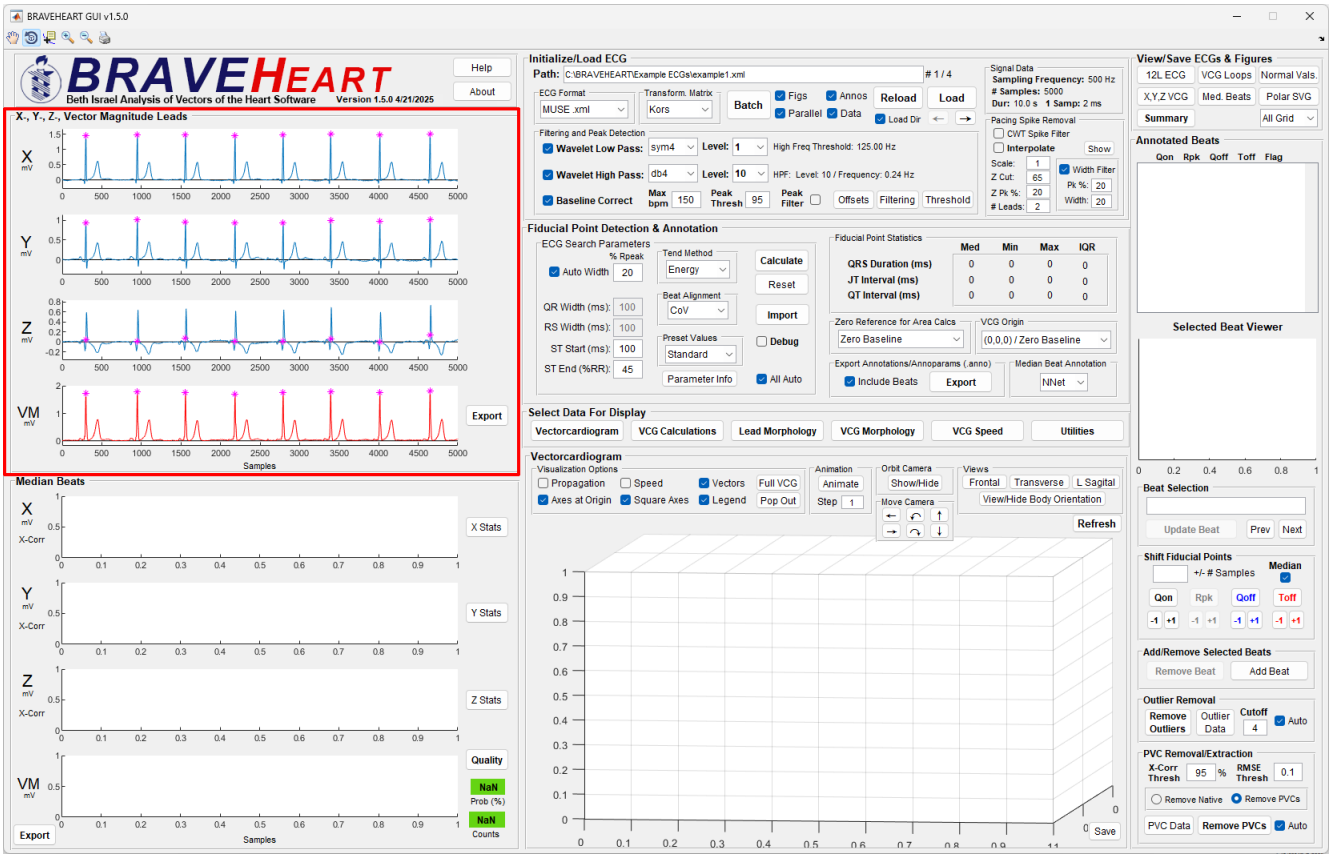
The '**Initialize/Load ECG**' section of the GUI controls loading ECG files, filtering, baseline correction, and transformation into a VCG.

Click the **Load** button:

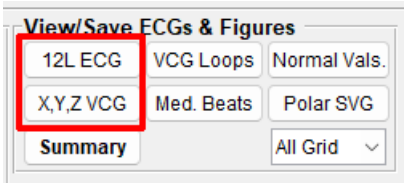


Select `example1.xml` from the appropriate directory and click 'Open' in the system dialogue box.

We see that the ECG has been loaded, converted into a VCG, and displayed in the upper left section of the GUI ('**X-, Y-, Z-, Vector Magnitude Leads**'):

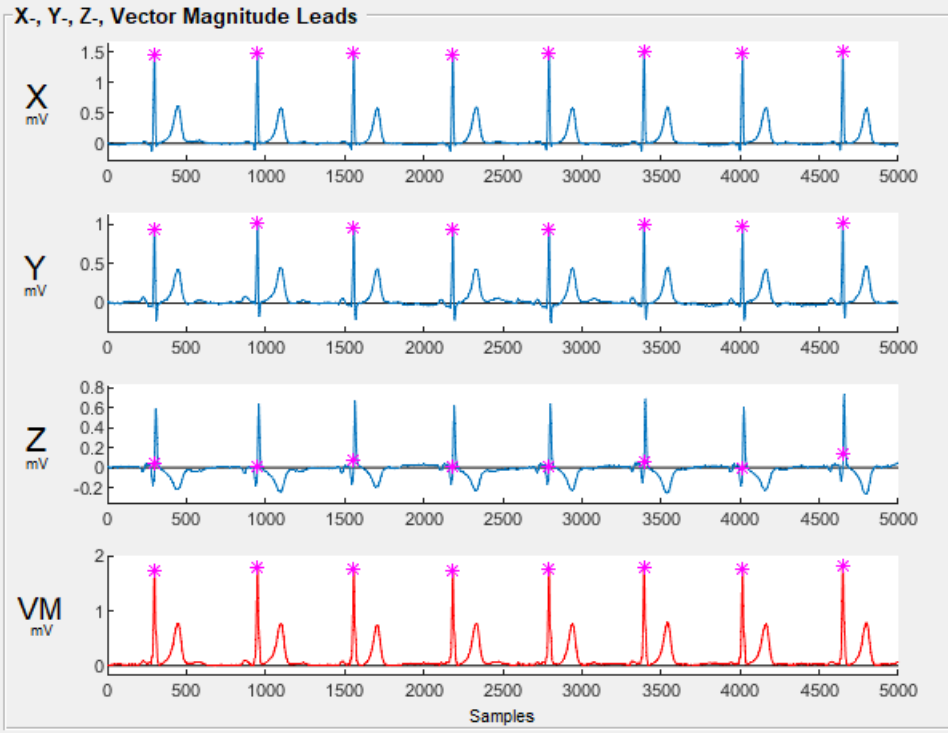


Once the ECG is loaded, we can view the standard 12-Lead ECG with 12 rhythm strips (12L ECG) and the VCG (X,Y,Z VCG) with rhythm strips by clicking on the appropriate buttons in the '**View/Save ECG & Figures**' section in the upper right (see **Chapter 12.8** and **Chapter 20**):

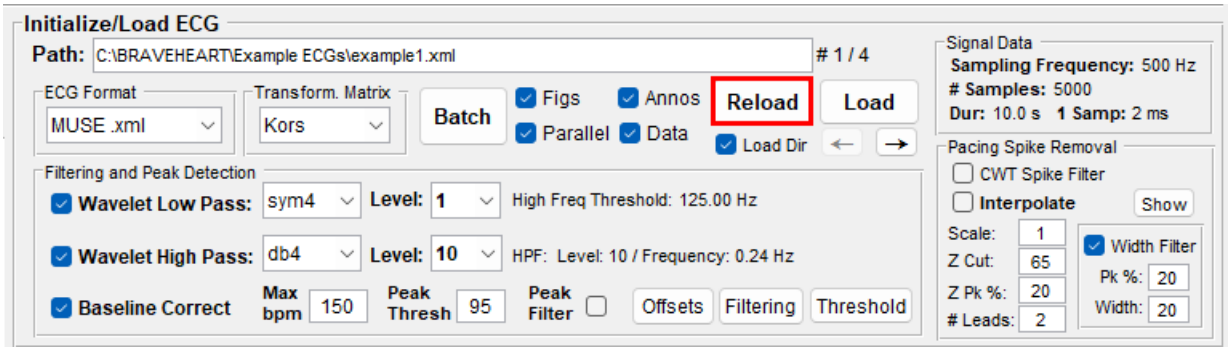


In the upper left section of the GUI ('**X-, Y-, Z-, Vector Magnitude Leads**'), note that each of the 8 R peaks has a magenta * which indicates the location of a detected R wave peak in the

vector magnitude ($VM = \sqrt{X^2 + Y^2 + Z^2}$) lead. In this example we are able to visually confirm that QRST complexes were not missed or over counted:

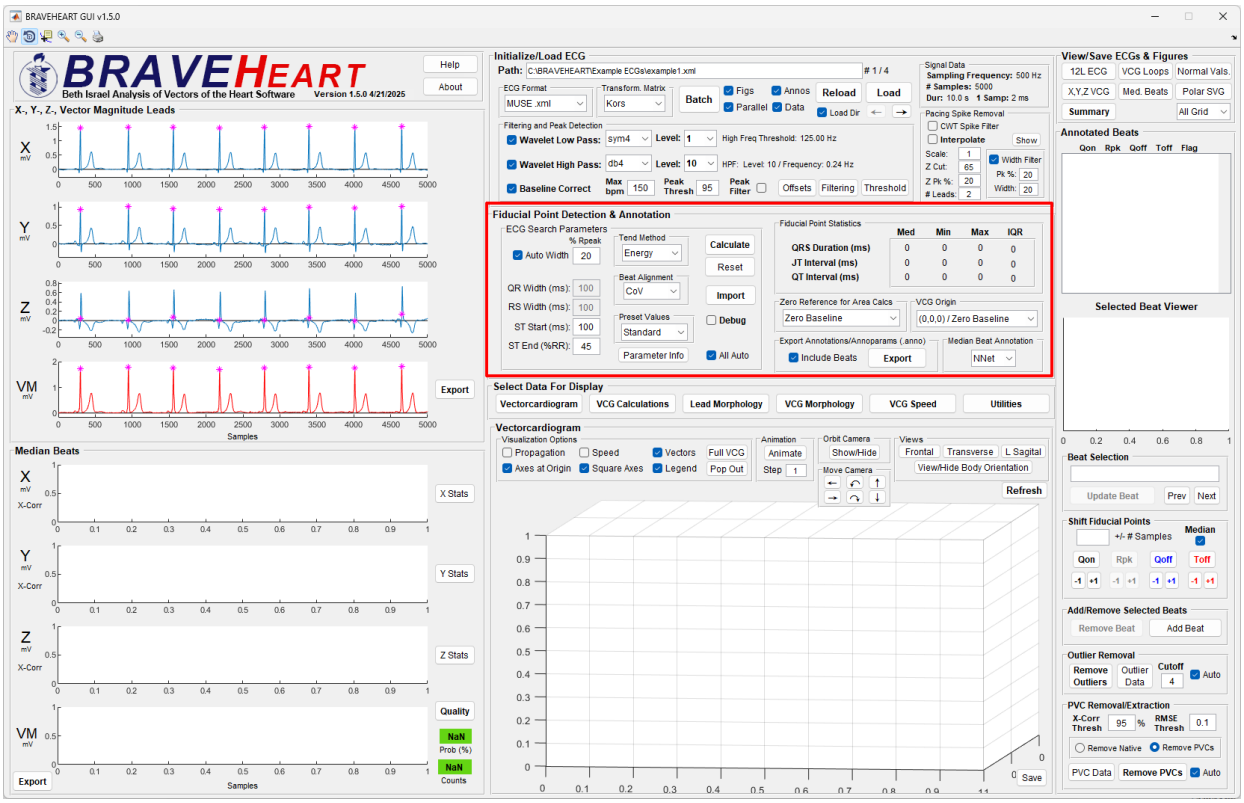


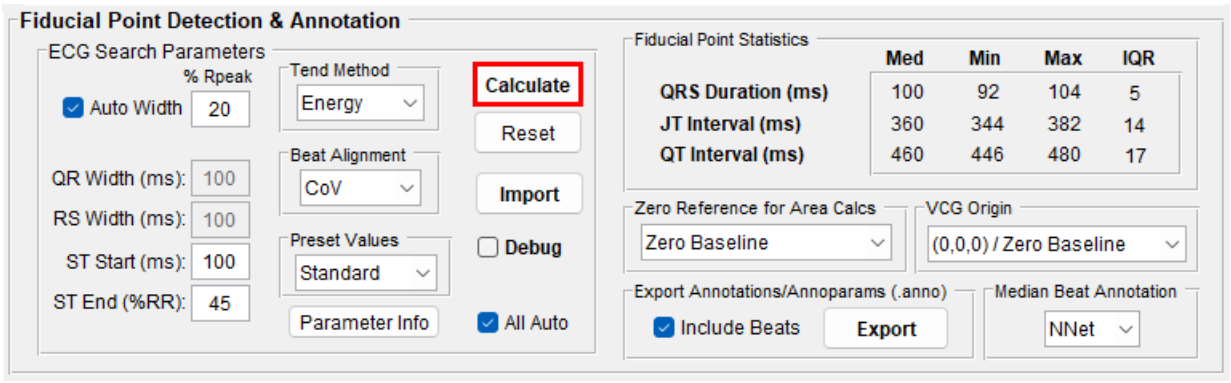
If we wanted to adjust ECG filtering parameters we would select the appropriate filtering parameters in the '**Initialize/Load ECG**' section (see **Chapter 6.2** and **Chapter 12.2**) prior to clicking **Load**. Alternatively we could change the parameters after the ECG is loaded and then click **Reload** which is to the left of the **Load** button:



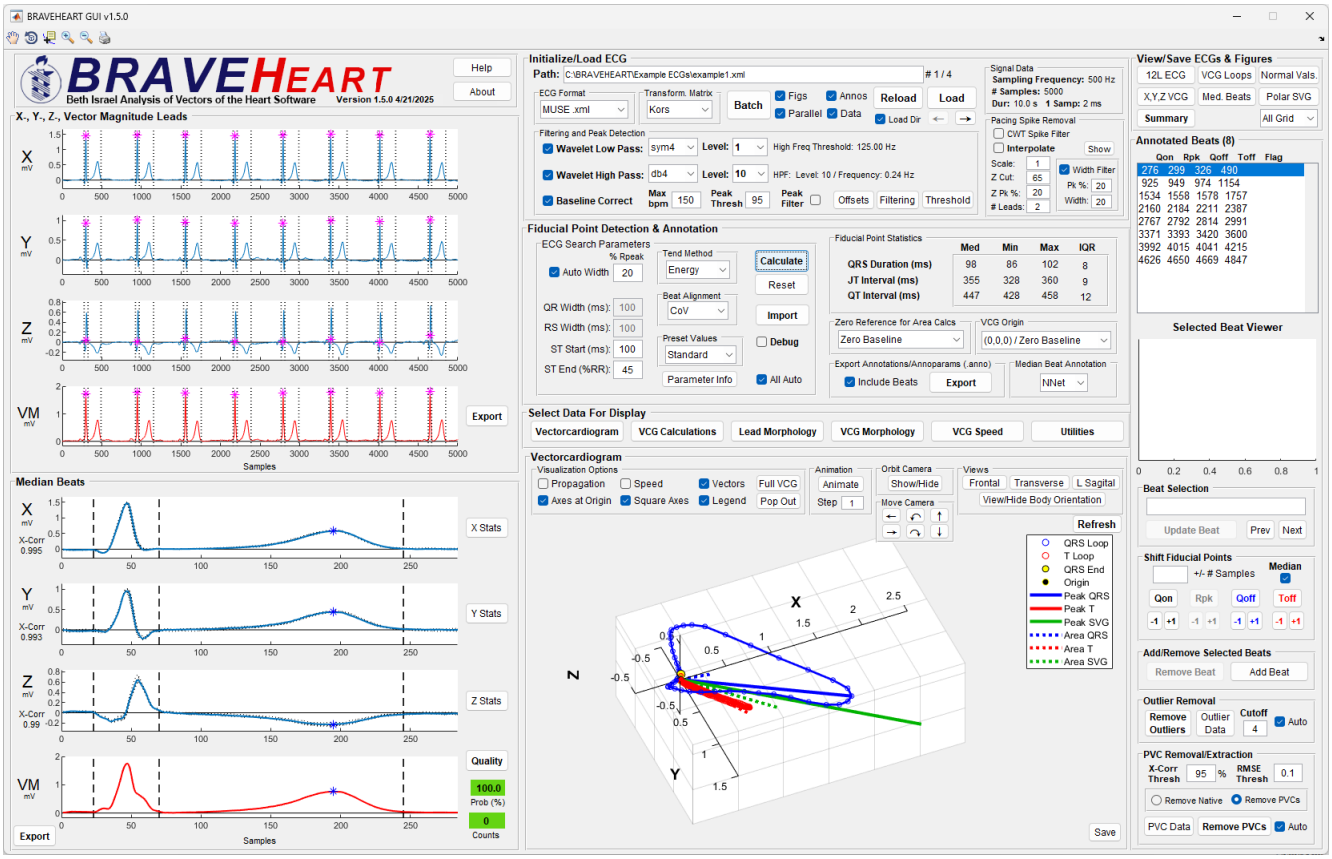
At this point the ECG has been loaded, filtered as appropriate, and converted into a VCG. We are now ready to further process the VCG with annotation, PVC/outlier beat removal, median beat construction, and calculations.

All of these processes can be performed by clicking the **Calculate** button in the '**Fiducial Point Dectection & Annotation**' Section:

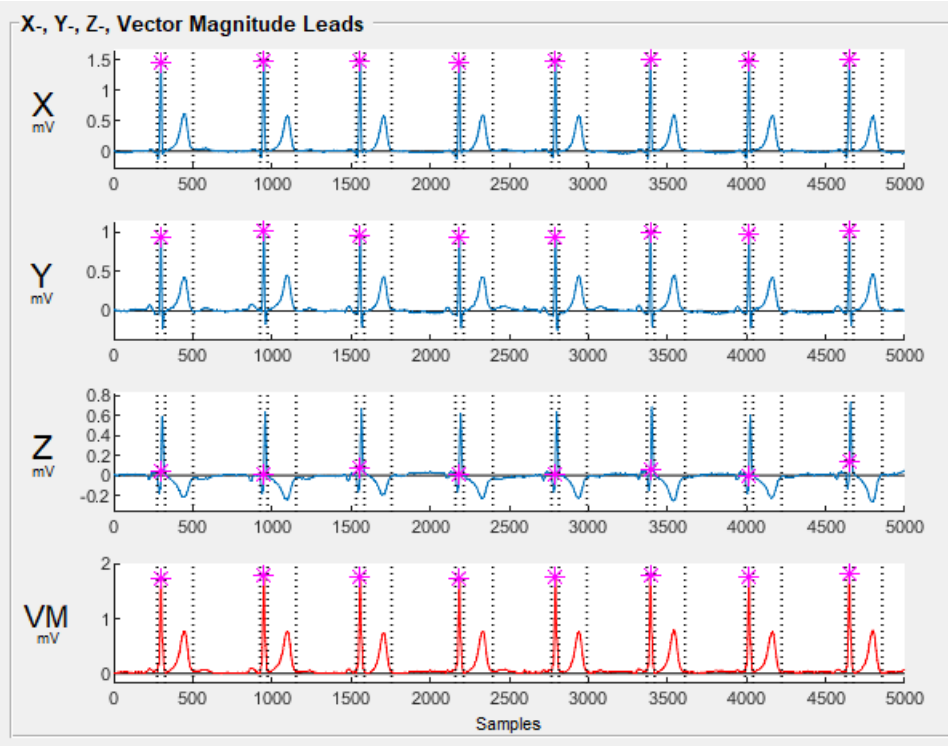




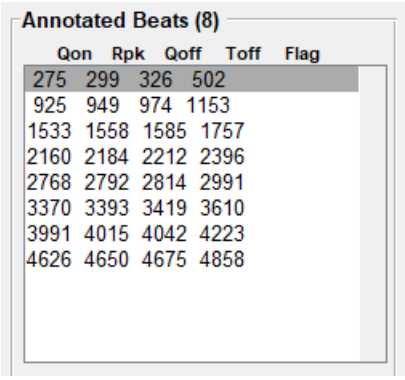
Click the **Calculate** button. The ECG is processed:



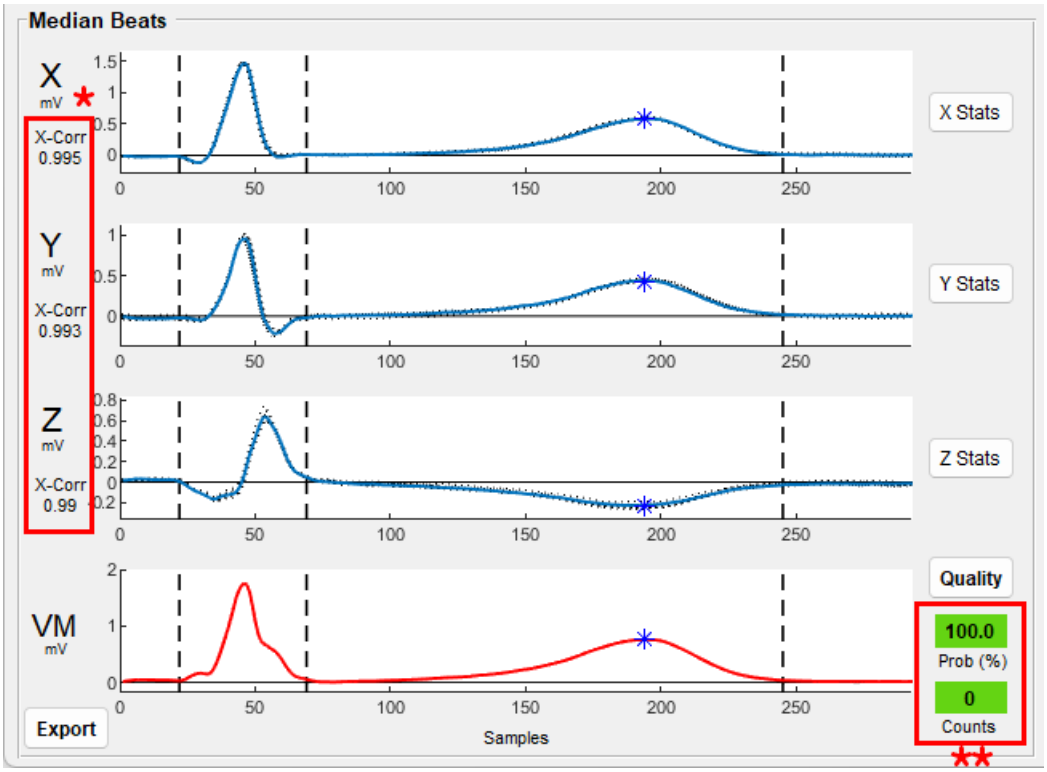
Note that after the ECG is processed using the **Calculate** button, new sections of the GUI have been filled in. For this ECG, no beats were detected as PVCs or outliers and removed (see **Chapter 15.4** and **Chapter 15.5**), and we see all beats continue to have a magenta *, indicating that they remain in the analysis. The '**X-, Y-, Z-, Vector Magnitude Leads**' section in the upper left now also shows the QRS onset (Q_{on}), QRS offset (Q_{off}), and T wave offset (T_{off}) annotations for each beat as dashed vertical lines:



Since the ECG has been annotated, on the right of the GUI, in the '**Annotated Beats**' section, we see the fiducial points (in samples – in the order of Q_{on}, R peak, Q_{off}, and T_{off}) which were used to segment out the 8 beats from the full ECG signals. These sample numbers correspond to the dashed lines and * in the '**X-, Y-, Z-, Vector Magnitude Leads**' section noted above.

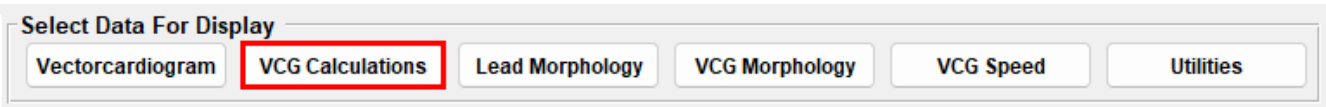


The median VCG beats (X, Y, and Z, and vector magnitude [VM]) are displayed in the lower left of the GUI in the '**Median Beats**' section:

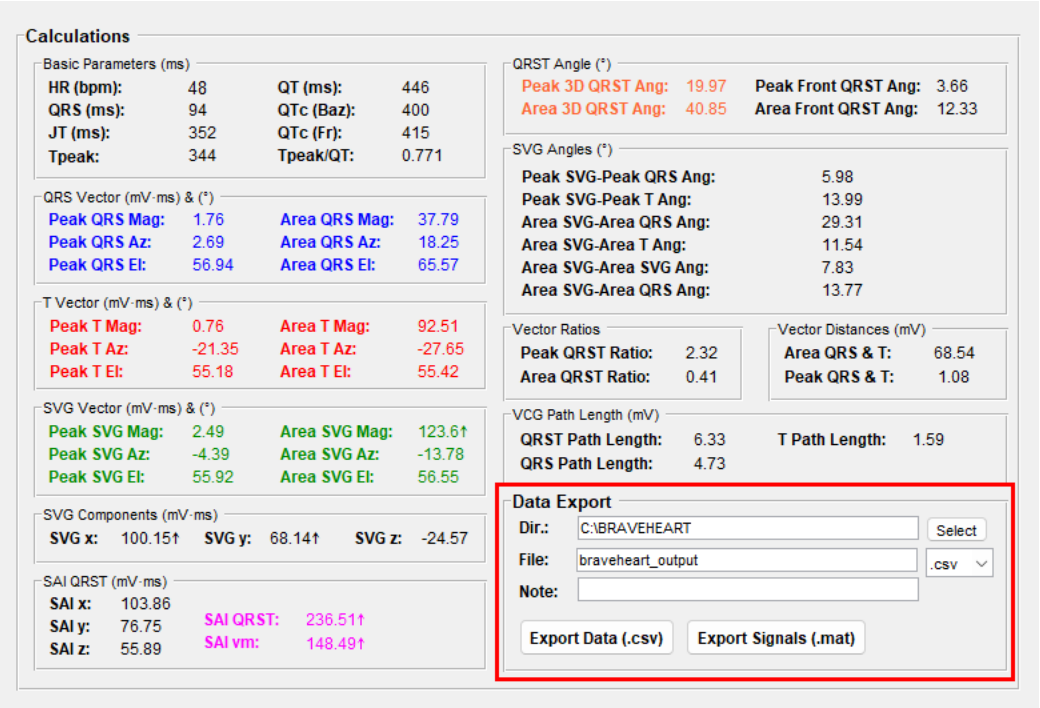


The individual beats that make up the X, Y, and Z median beats are displayed as black dotted plots, and the median beat fiducial points Q_{on}, Q_{off}, and T_{off} are denoted with vertical black dashed lines. This section also allows assessment of ECG processing quality. Note that the normalized cross correlation ('X-Corr') values for the X, Y, and Z leads on the left (red box with *) are all very close to 1, indicating that all of the beats that were used to construct the median beat are similar and aligned property. The 'Prob (%)', in the lower right (red box with **), which is the predicted probability of the processed ECG being "good" quality, is 100%, and the number of poor quality flags ('Counts') is 0 indicating good quality processing. For further details of ECG quality assessment see **Chapter 16**.

Click on the **VCG Calculations** button in the middle of the GUI which displays calculations from the `VCG_Calc` results class (see **Chapter 24**):



This brings up the '**VCG Calculations**' section which displays various VCG measurements and is also used for exporting all calculations (including parameters which are not displayed in the GUI). The controls for data export are shown in the bottom right:



The default directory used to save files is the same directory from which the ECG was loaded. To change the directory for saving click Select and choose a new directory. The default file name for the saved data file is `braveheart_outut`; change this to what you want to name the output file.

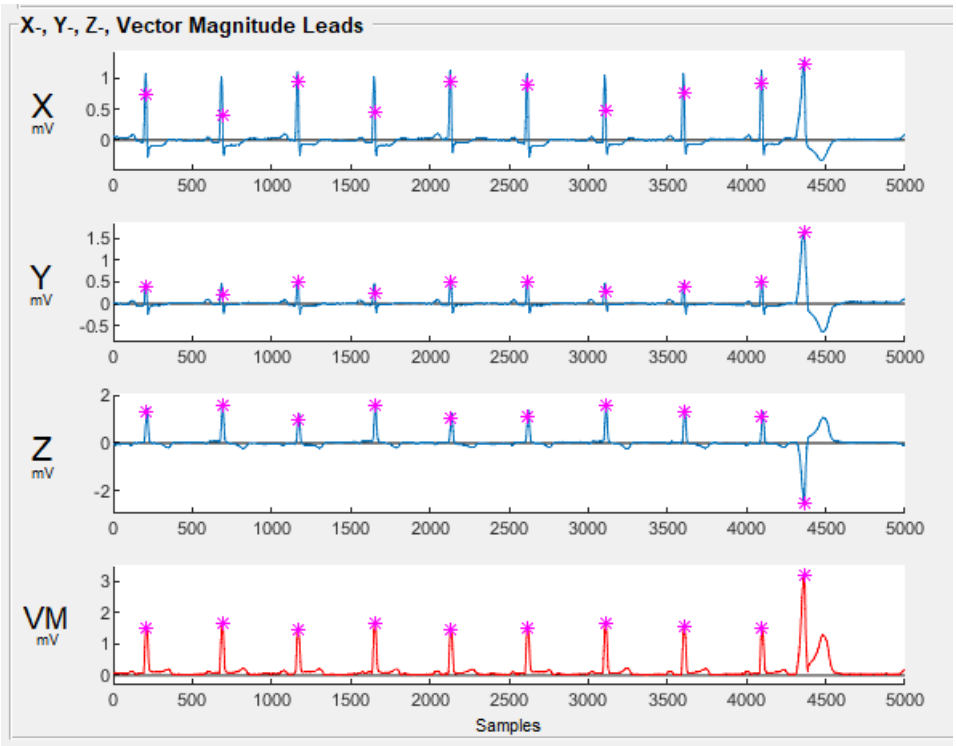
Once your file name and directory are set, click **Export Data (.csv)** to export the data to a `.csv` file located in the directory chosen above and with the specified file name. Once the file is successfully exported, **Success!** is displayed in green in the bottom right of the section.

5.2 Example 2 - ECG with PVCs

In this example we will load an ECG with a PVC, and show different ways of removing the PVC (see **Chapter 15.4** for additional information on PVC detection/removal).

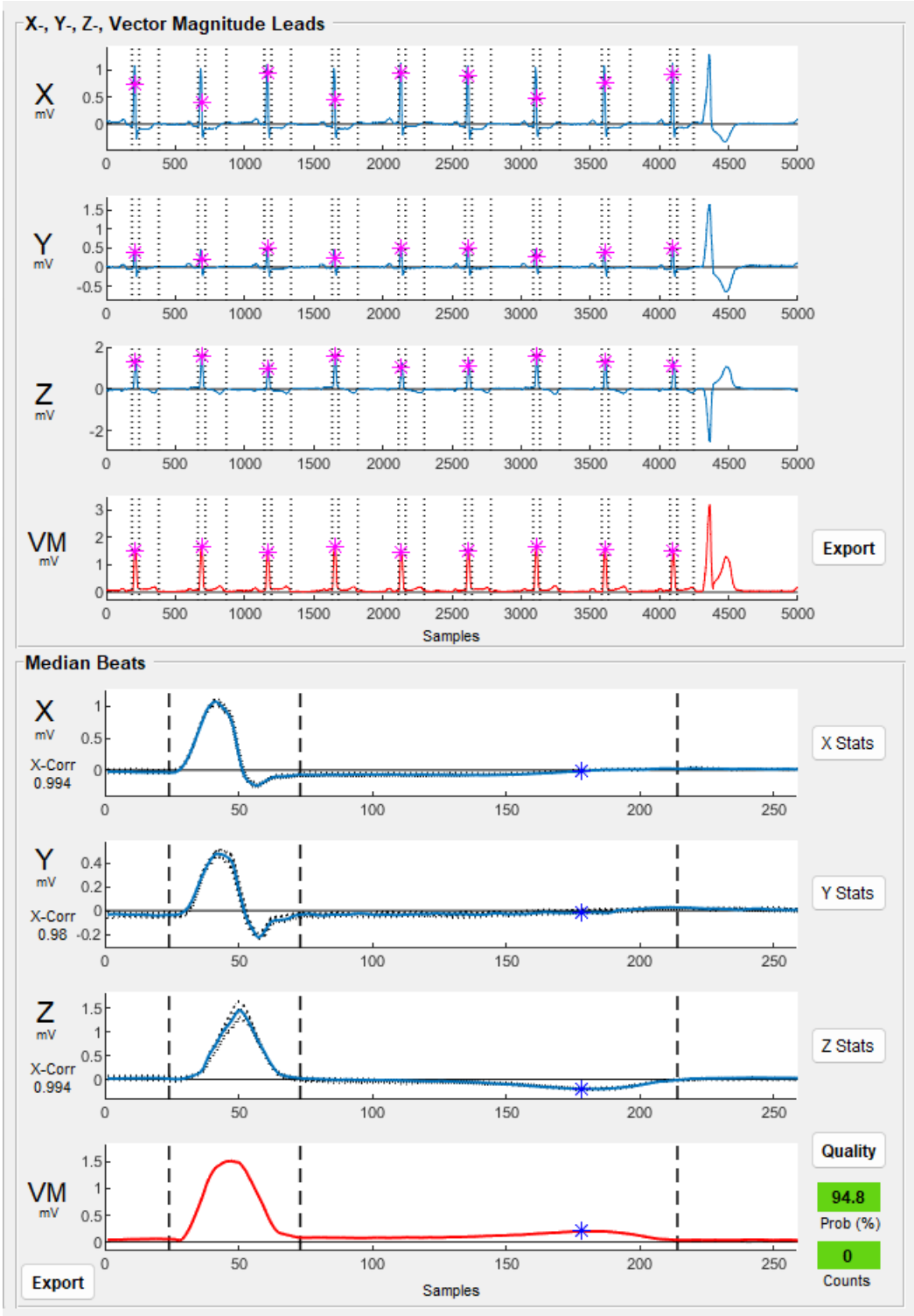
First load `example2.xml` using the **Load** button as noted above in Example 1 (**Chapter 5.1**).

By inspecting the '**X-, Y-, Z-, Vector Magnitude Leads**' section, note that the last beat of this ECG is a PVC, and all beats/R peaks are correctly identified with a *:



Click the **Calculate** button:

The ECG is processed, and because automatic PVC removal was turned on (this is enabled by default), the PVC is automatically removed. We can see the PVC has been removed by looking at the '**X-, Y-, Z-, Vector Magnitude Leads**' section where we see that PVC does not have a * anymore, the fiducial points for the PVC are not present in the **Annotated Beats**' section, and the PVC is not seen in the '**Median Beats**' section:

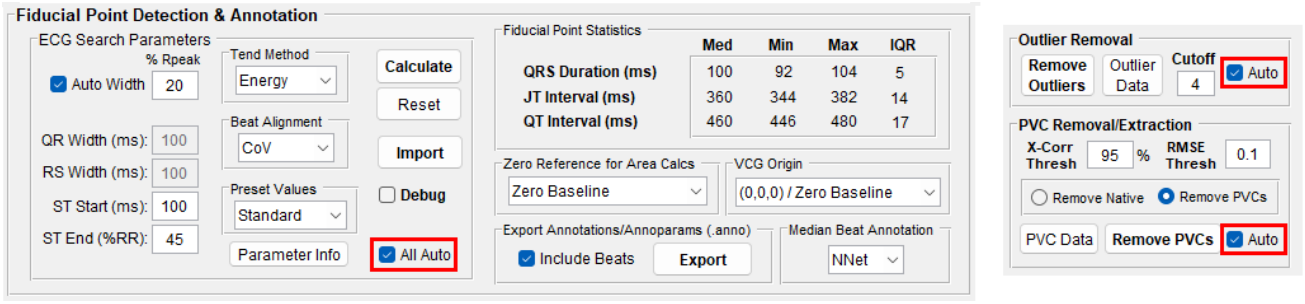


Also note that the Quality metrics (green text boxes in the lower right of the '**Median Beats**' section show an excellent estimated quality ('Prob (%)') of approximately 95% (see **Chapter 16**).

Now click the **Reload** button:

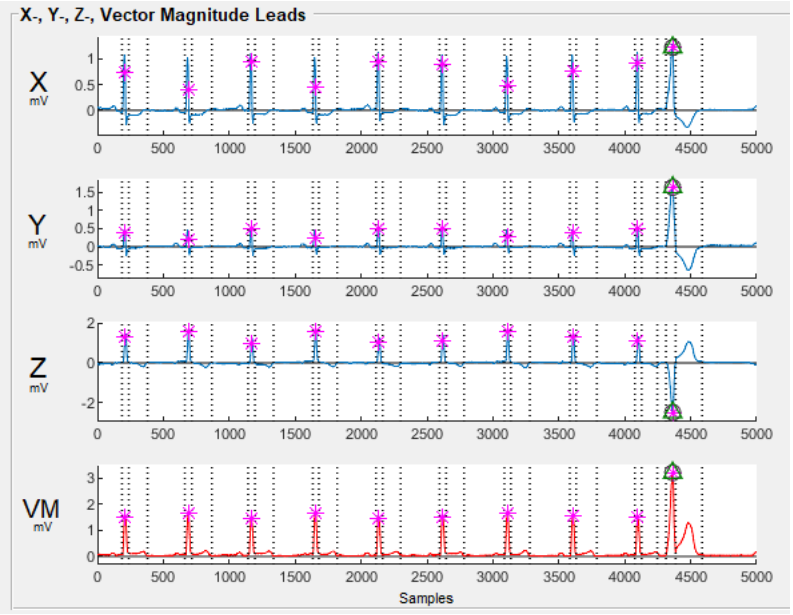
The GUI is cleared and the ECG is reloaded without having to select the file again.

This time we will disable automatic PVC removal. We can do this by either unchecking ☑ All Auto in the '**Fiducial Point Detection & Annotation**' section which disables BOTH PVC and outlier removal, or by unchecking ☑ Auto in the PVC removal section in the bottom right of the GUI.



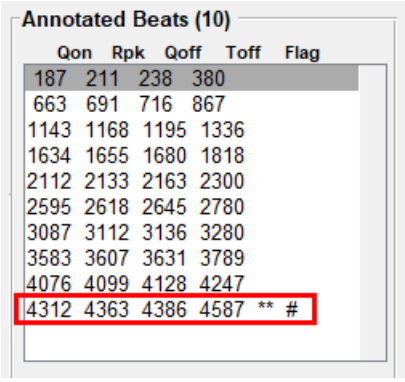
If you uncheck ☐ All Auto, the 2 ☐ Auto checkboxes in the PVC and outlier removal sections also become unchecked. Similarly, checking ☑ All Auto checks both ☑ Auto checkboxes.

Uncheck ☐ All Auto (and note that the 2 ☐ Auto checkboxes in the '**PVC Removal/Extraction**' and '**Outlier Removal**' sections also become unchecked), and then click Calculate. This time the PVC is not removed; note that in the '**X-, Y-, Z-, Vector Magnitude Leads**' section, the PVC still has a * and annotations (dashed lines):

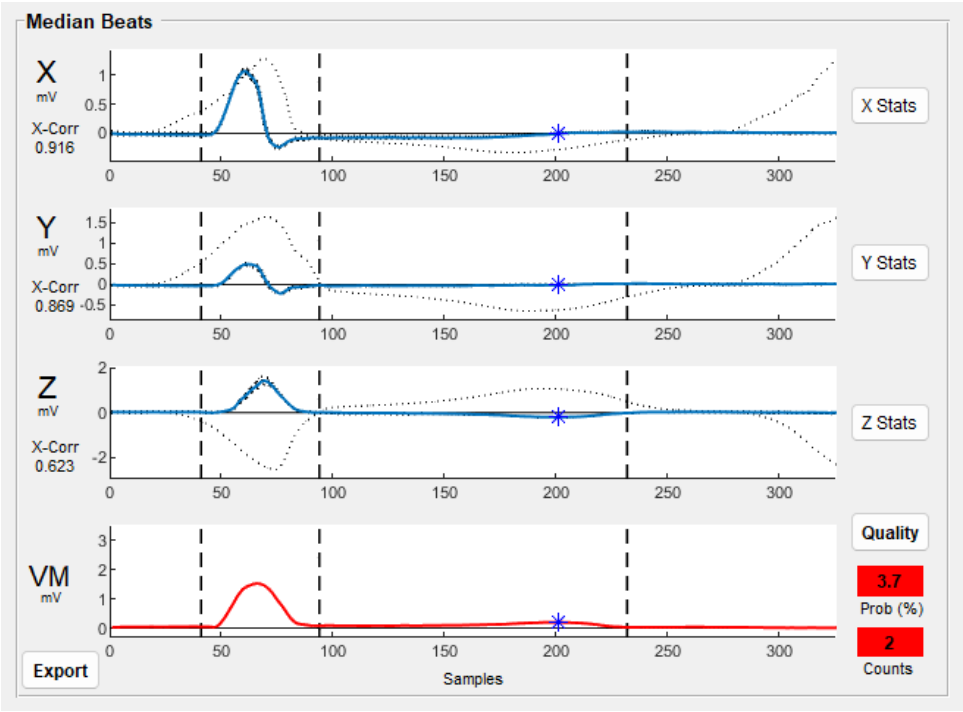


Also note that the PVC has a triangle around the * (⚠) which is how a beat that has been detected as a PVC is marked. In this case there is also a circle around the * because the PVC is additionally detected as an outlier (see Example 3 in **Chapter 5.3** and **Chapter 15.5**).

As the PVC is still present, the PVC fiducial points are also still present in the **Annotated Beats**' list where the PVC's fiducial points are followed by a '#', which is used to signify that the beat has been detected as a PVC by BRAVEHEART's PVC detection algorithm (the '**' denotes that this beat was classified as an outlier as well):

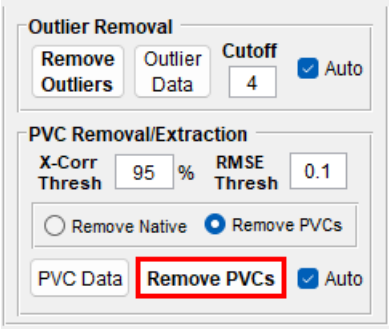


In the '**Median Beats**' section the PVC is represented by a black dotted line that is different from the other beats:



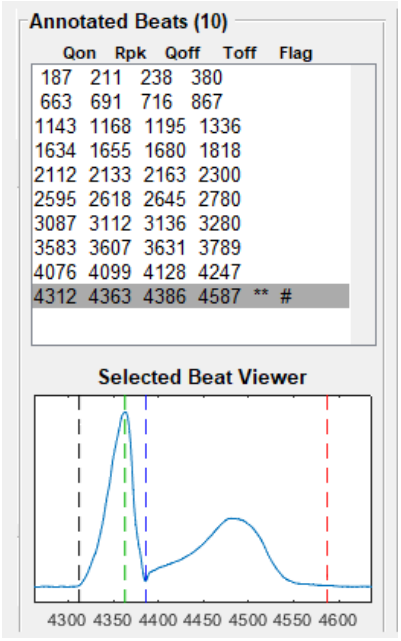
Also note that the quality metrics (red text boxes in the lower right of the '**Median Beats**' section shows a low probability of "good" quality 'Prob (%)') of only 3.7% due to the PVC (see **Chapter 16**).

We will first remove the PVC using the PVC removal algorithm. In the '**PVC Removal/Extraction**' section in the bottom right of the GUI, simply click **Remove PVCs**:

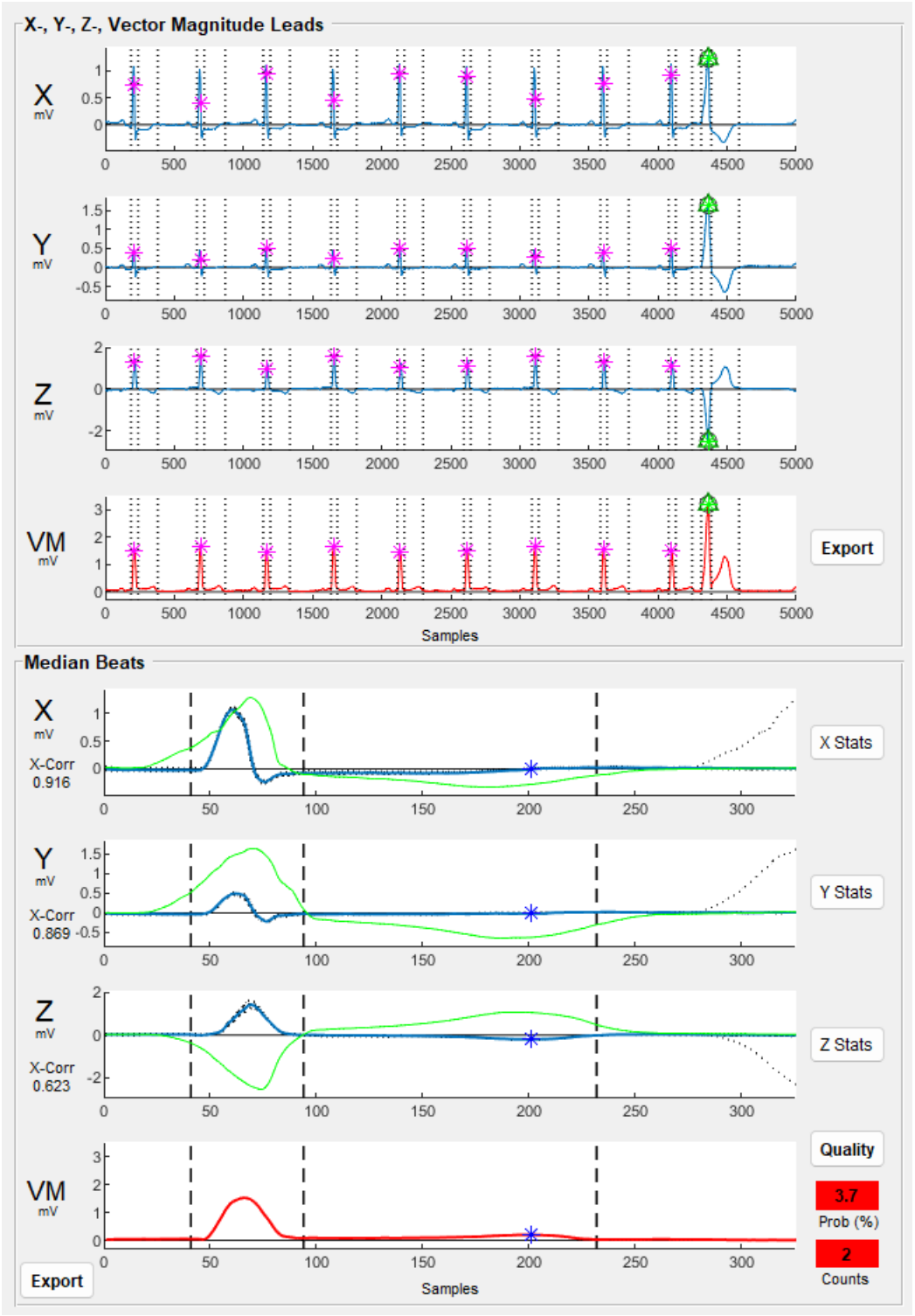


The ECG is reprocessed, the PVC is removed, and the median beat is recreated without the PVC.

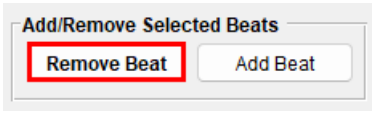
Alternatively we could manually tell BRAVEHEART to remove the PVC. Reload the ECG with the **Reload** button. Click on the PVC beat in the '**Annotated Beats**' list. The selected beat is highlighted in the list and appears in the '**Selected Beat Viewer**' right below the '**Annotated Beats**' list:



The selected beat is also highlighted with a green * in the '**X-, Y-, Z-, Vector Magnitude Leads**' section, and the beat is highlighted in green in the '**Median Beats**' section:



Click on the **Remove Beat** button in the '**Add/Remove Selected Beats**' section:



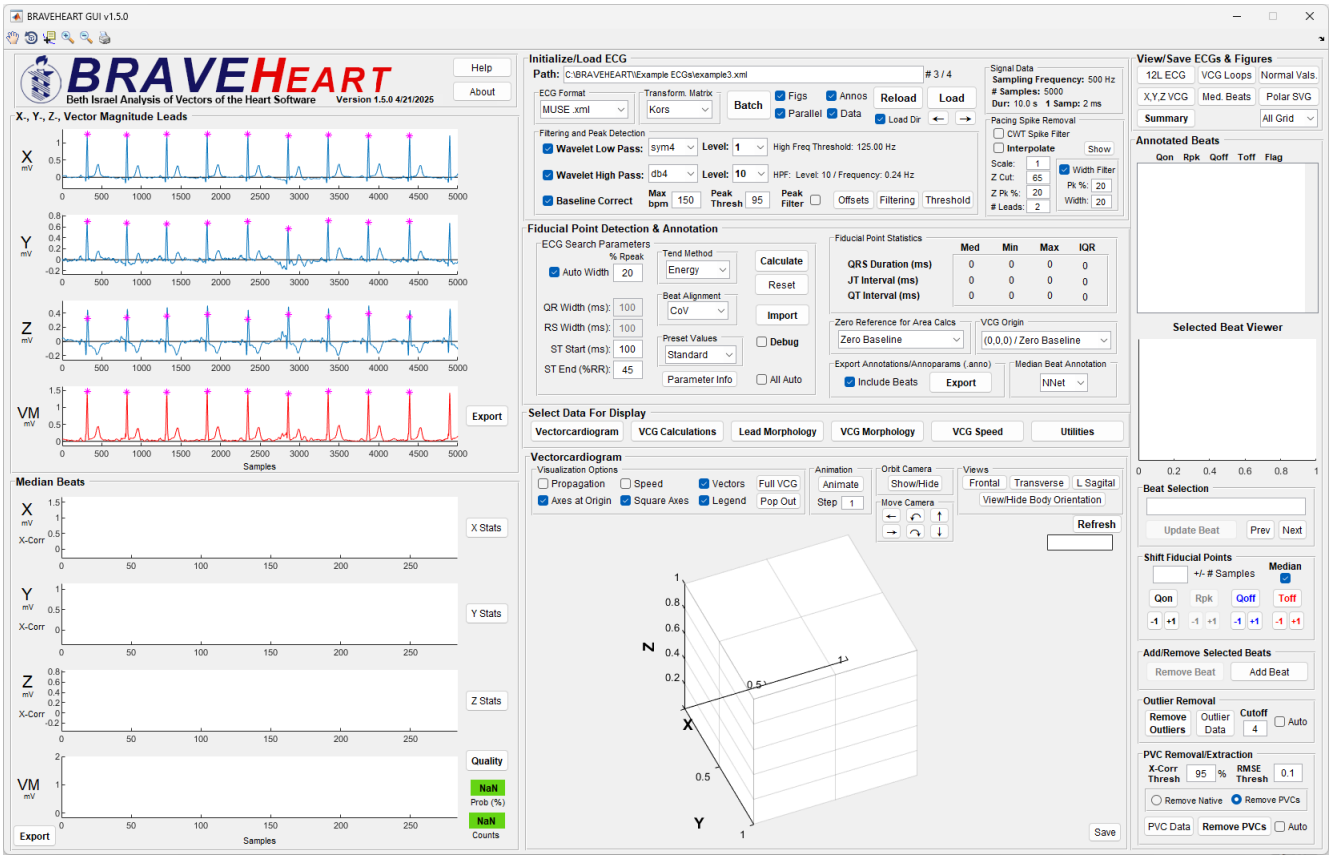
The ECG is reprocessed, the PVC is removed, and the median beat is recreated without the PVC. Once the PVC is removed, the estimated quality of the ECG is now high (94.8 % with green color background).

At this point, the data can be exported as noted above in Example 1 (**Chapter 5.1**).

5.3 Example 3 - ECG with Artifact

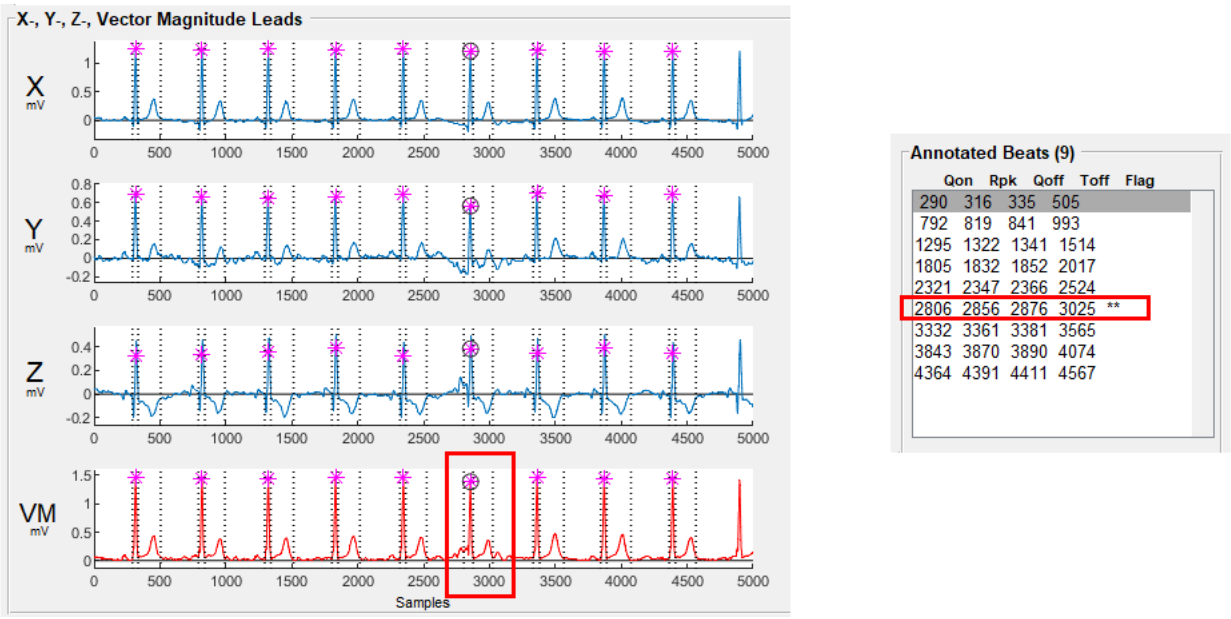
In this section we will load an ECG with focal noise that is removed with outlier detection and removal.

First load `example3.xml` and disable ☐ All Auto as noted above in Example 2 (**Chapter 5.2**).

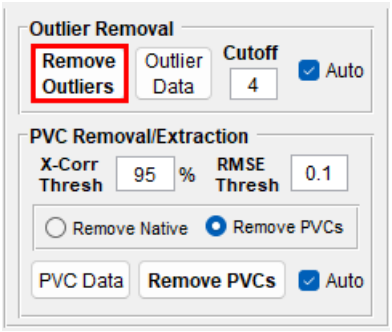


Note that the 6th beat has some focal noise.

Click the **Calculate** button. The 6th beat is identified as an "outlier" as noted by the circle around the * (⊛), and the '**' next to the beat in the '**Annotated Beats**' section.



Had we had ☑ All Auto or the ☑ Auto in the '**Outlier Removal**' section checked off, this beat, identified as an outlier, would have been automatically removed. In this case, however, because we disabled automatic outlier removal, we will have to manually remove the outliers similar to how PVCs are removed in Example 2 (**Chapter 5.2**). The outliers can be automatically removed by clicking the **Remove Outliers** button:

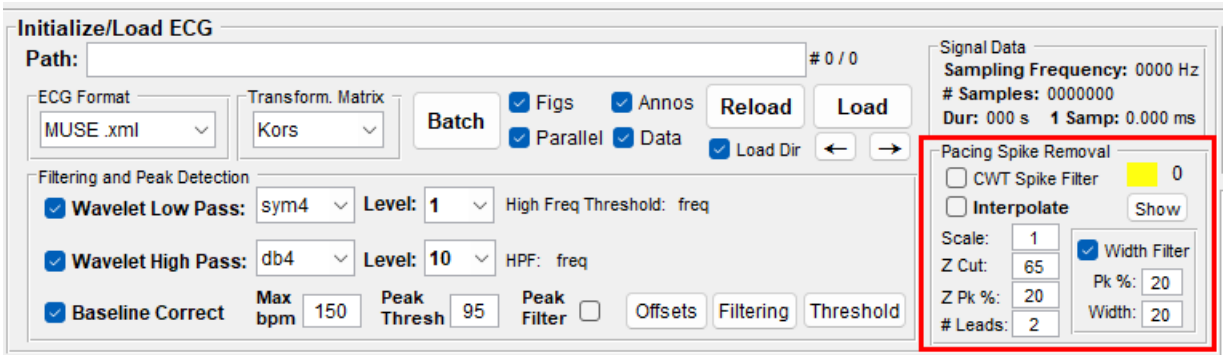


The outlier can also be removed by clicking on the beat in the '**Annotated Beats**' section and then clicking the **Remove Beat** button in the '**Add/Remove Selected Beats**' section as noted in Example 2 (**Chapter 5.2**).

5.4 Example 4 - Dealing with Pacing Artifact

In this example we will see how BRAVEHEART deals with pacing artifact. Note that as of version 1.5.0 there are now 2 ways to deal with pacing spikes. These methods are described in detail in **Chapter 15.2**.

Pacemaker artifact detection and removal are controlled in the bottom right of the '**Initialize/Load ECG**' section of the GUI:

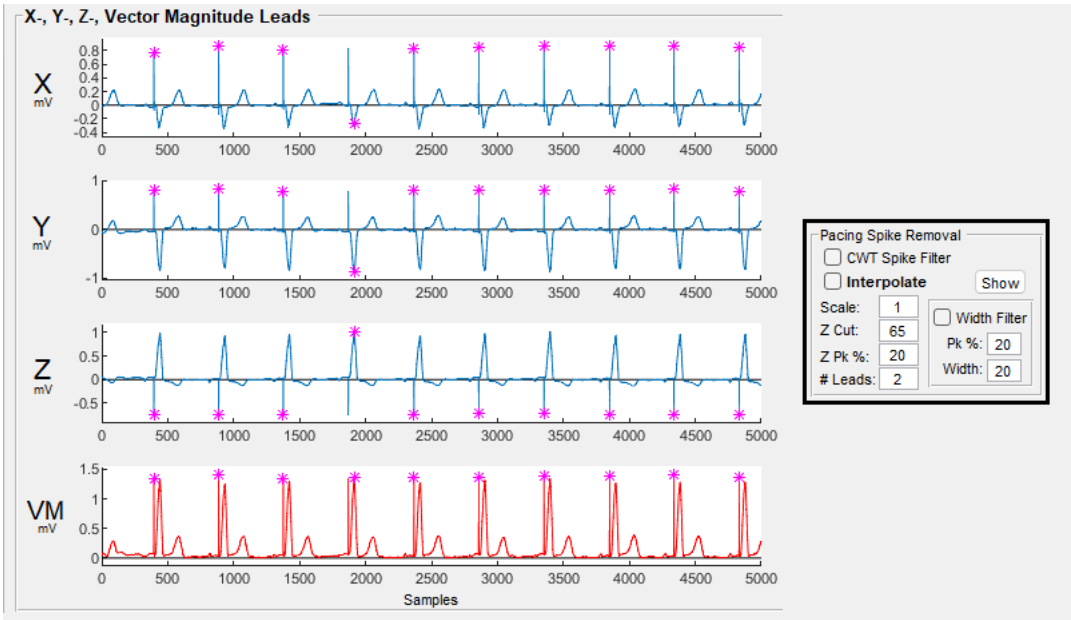


The ☑ Width Filter checkbox controls the original method of pacemaker spike detection, and the ☐ CWT Spike Filter checkbox controls the new method which utilizes the continuous wavelet transform (CWT). In the GUI it will not let you select both of these methods at the same time.

We will start by using the original method using the "median filter". This is the default setting because it is significantly faster than using the "CWT method".

Under the ☑ Width Filter checkbox, the 'Pk %' and 'Width (ms)' textboxes usually can stay at their nominal values of '20' unless the pacing spikes are very large, wide, or complex. Further details on these parameters are available in **Chapter 6.2**. Since we are using the median filter, the other parameters related to the CWT method ("Scale", "Z Cut", "Z Pk %", and "# Leads") do not control any part of the ECG processing.

We will start by disabling pacemaker artifact removal by unchecking the ☐ Width Filter checkbox. Now both the ☐ Width Filter and ☐ CWT Spike Filter checkboxes are unchecked. Next, load `example4.xml` as noted in the above examples:



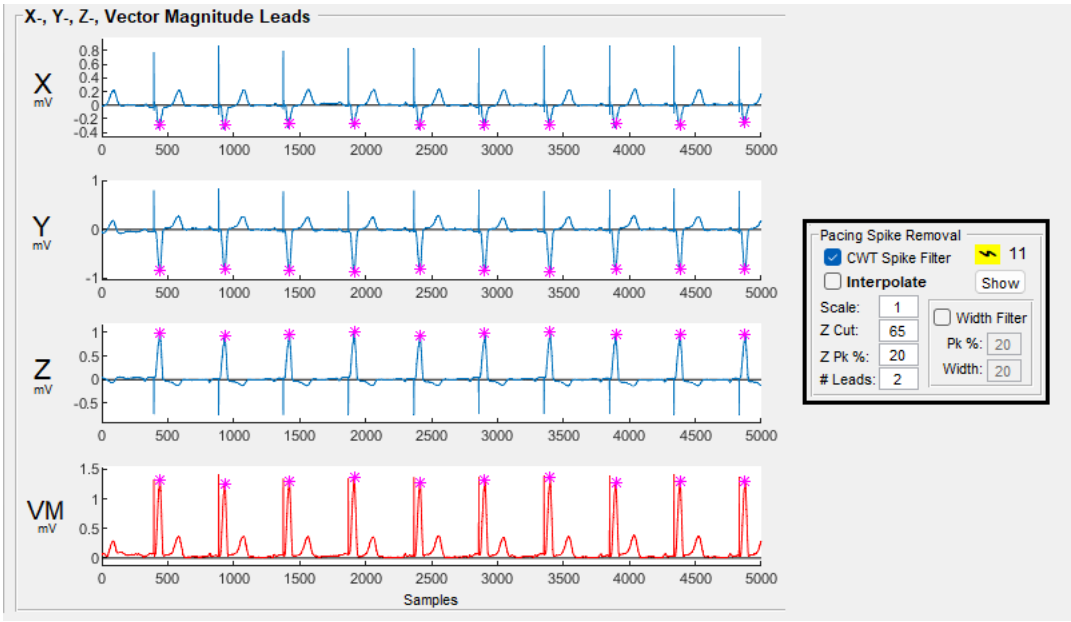
The '**X-, Y-, Z-, Vector Magnitude Leads**' section shows that for almost all of the beats the R peak has been detected as the large pacing spike (see the location of the *) because in most beats the pacing artifact magnitude is larger than the QRS complex magnitude in the VM lead. This is very problematic, as if you press **Calculate** now you will see that the ECG does not process correctly and the results are unusable.

Re-enable pacemaker artifact removal by checking the ☑ Width Filter checkbox, and then reload the ECG with the **Reload** button:

This time, as pacemaker artifact removal was enabled, the pacemaker spikes are ignored, and the true QRS peaks are detected as the R peaks. If you press **Calculate** the ECG will process correctly.

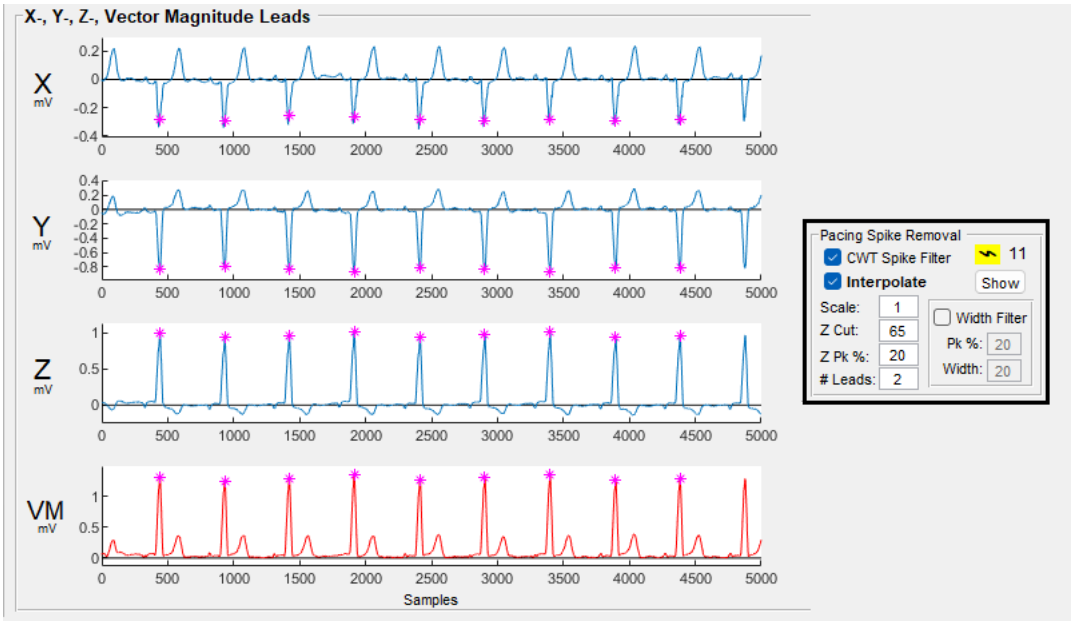
Now we will use the new CWT method method which was added in version 1.5.0. This new method is significantly more robust than the legacy median filter method, and it can more effectively process very large or complex pacing spikes. However, the downside of this method is that is is significantly slower than the median filter method. For further details see **Chapter 15.2**.

To start we will reload the ECG, but this time with the ☑ CWT Spike Filter checkbox activated and the nominal settings for 4 parameters "Scale", "Z Cut", "Z Pk %", and "# Leads". Further details on these parameters are available in **Chapter 6.2**. Checking the ☑ CWT Spike Filter checkbox will automatically uncheck the ☐ Width Filter checkbox.



This time we see that the result is identical to the prior example when we used the median filter method. Note that there is a small lightning bolt in a yellow square next to the number 11 indicating that pacing spikes were detected in 11 out of 12 leads.

We now will use another useful function of the CWT method, which is removing the actual pacing spikes via interpolation. We will now check the ☑ **Interpolate** checkbox under the ☑ CWT Spike Filter checkbox. Now reload the ECG with the **Reload** button:

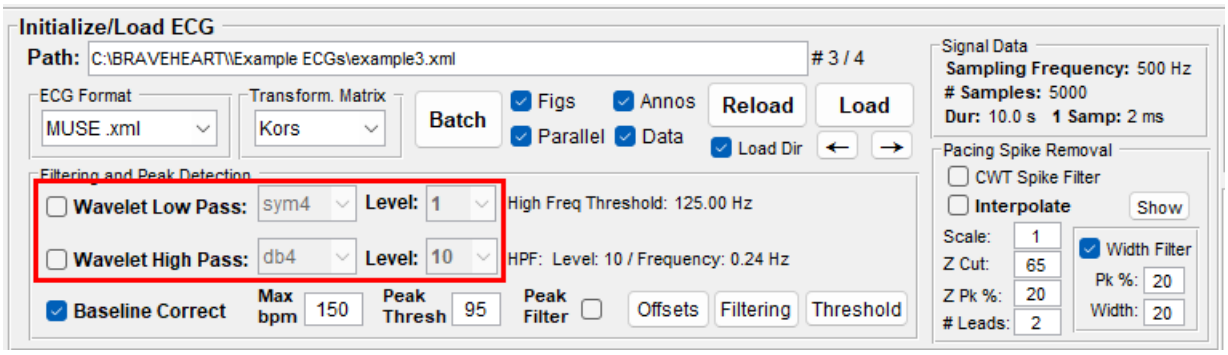


Now the R peaks are correctly identified, and the pacing spikes have been removed from the processed ECG. If you press the Show button details of how the ECG pacing spikes were identified and removed via interpolation can be views. See **Chapter 15.2** and **Chapter 20** for further details.

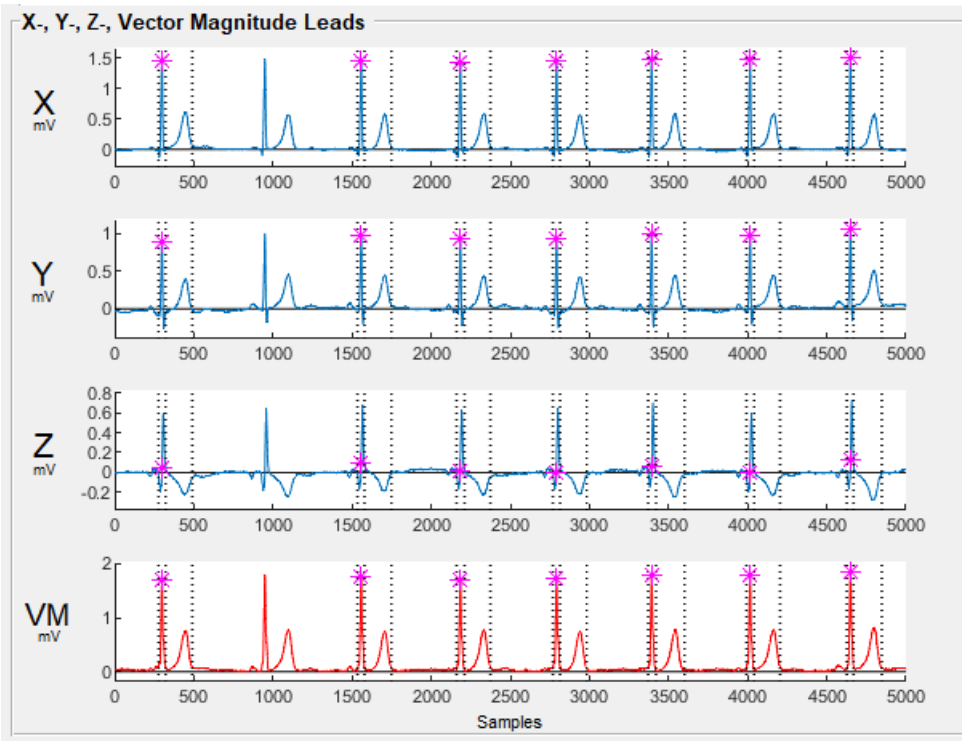
5.5 Example 5 - Using `.anno` Files

In this example we will load an ECG, process with non-nominal settings, remove some beats, and then use an `.anno` file to reproduce this work-flow automatically the next time the ECG is loaded.

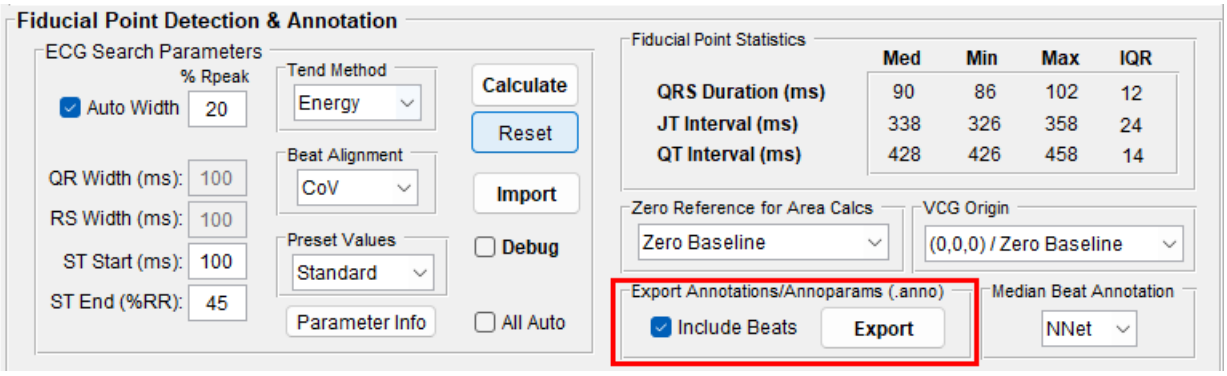
For illustrative purposes we will disable filtering and remove the second beat. Uncheck ☐ **Wavelet Low Pass** and ☐ **Wavelet High Pass** in the '**Initialize/Load ECG**' section:



Next, load `example1.xml` and click **Calculate** as noted in the examples above. Click on the second beat in the '**Annotated Beats**' listbox, and then delete the second beat using the **Remove Beat** button. The second beat is removed from the median beat and calculations:



Note that the * is not present on the second beat indicating that the second beat is no longe included in the median beat. Now click on the **Export** button in the '**Fiducial Point Detection & Annotations**' section of the GUI, making sure that the ☑ Include Beats checkbox is checked (this saves the beat annotations in addition to the anotation parameters).

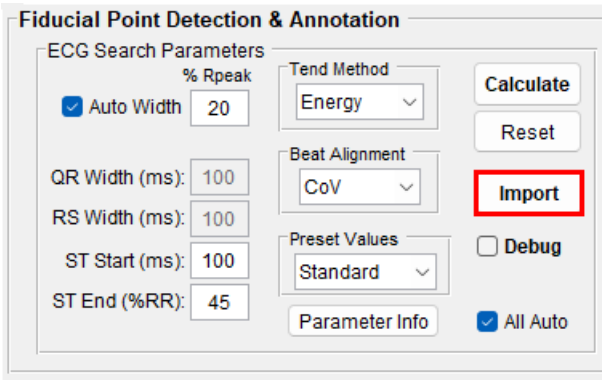


A file named `example1.anno` appears in the same folder as `example1.xml`.

Reset all GUI parameters to nominal values with the **Reset** button which restores the default

GUI/annotation parameters (see **Chapter 6**) and re-checks ☑ **Wavelet Low Pass** and ☑ **Wavelet High Pass**. Then reload the ECG from scratch with the **Reload** button.

This time, instead of clicking the **Calculate** button to process the ECG, we will reproduce what we did above by loading the `.anno` file: Click **Import** and choose `example1.anno`:

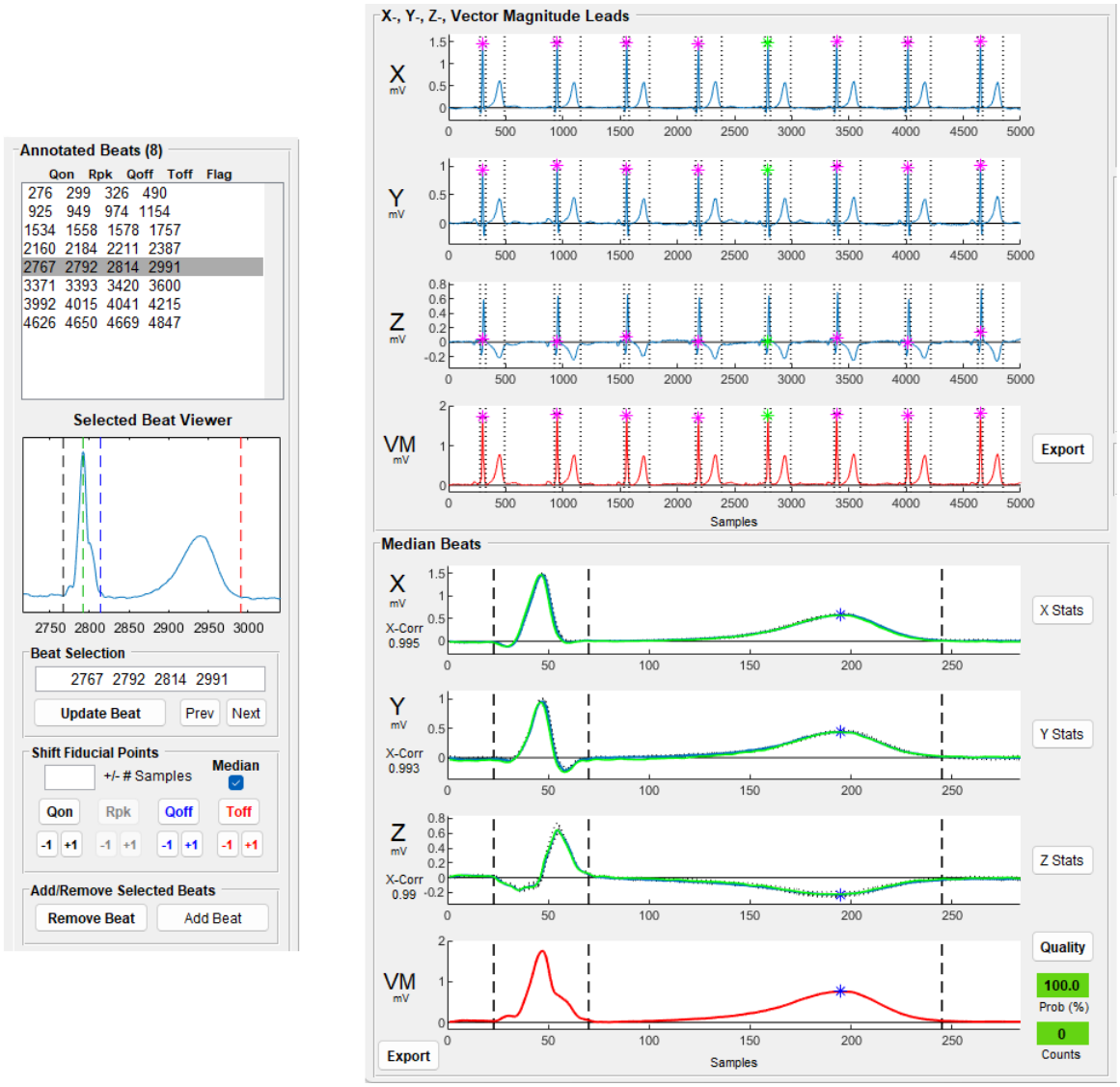


`example1.xml` is reloaded using the annotation/filtering parameters specified in the `.anno` file. Note how ☐ **Wavelet Low Pass** and ☐ **Wavelet High Pass** are now unchecked, and beat 2, which was manually deleted before saving the `.anno` file, has been automatically deleted.

Using the **Import** button is equivalent to placing the file `example1.anno` in the same folder as the original ECG file `example1.xml` and then running batch ECG processing – the parameters/beats in the `.anno` file overwrite the default parameters and beat removal. Note that when using `.anno` files, after importing, automatic PVC and outlier removal are disabled.

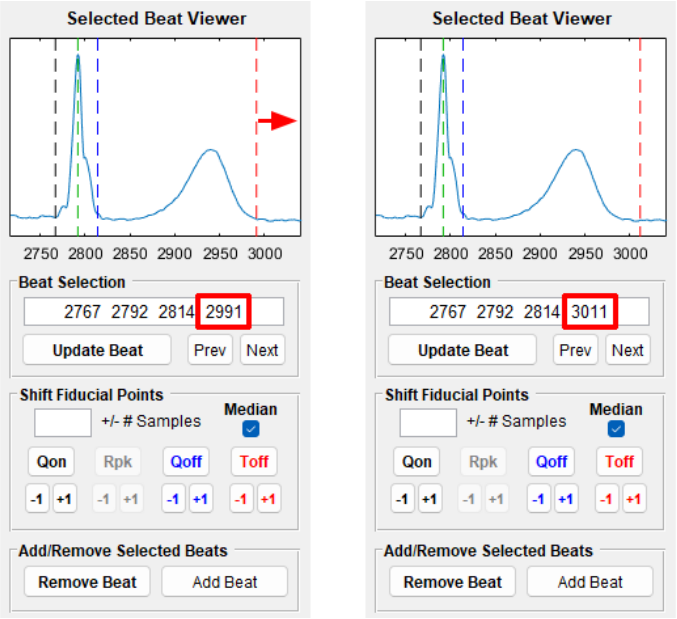
5.6 Editing Beats

The '**Annotated Beats**' listbox contains fiducial point annotations for Q_{on}, R peak, Q_{off}, and T_{off} for each individual beat that was used to create the median beat. To edit a beat's annotations, click on the row which corresponds to the beat in the listbox. The selected beat appears in the '**Selected Beat Viewer**', the R peak of the selected beat is highlighted with a green * in the '**X-, Y-, Z-, Vector Magnitude Leads**', and the selected beat is highlighted in green in the '**Median Beats**' section. Below the '**Selected Beat Viewer**' there is a '**Beat Selection**' section that shows the fiducial points from the '**Annotated Beats**' listbox. In the figure below the 5th beat has been selected:

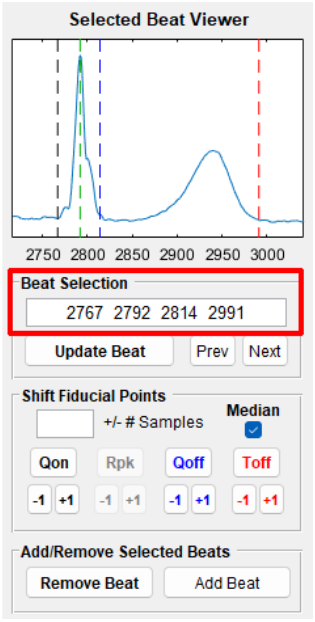


To edit the beat's fiducial points there are a few options:

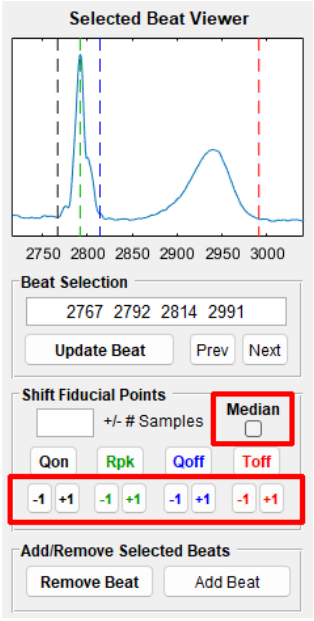
1. Drag the dashed lines on the '**Selected Beat Viewer**' to their appropriate locations. Black is Q_{on}, green is R peak, blue is Q_{off}, and red is T_{off}. Once the mouse is released, the new location of the fiducial point is updated in the '**Beat Selection**' textbox. In the figure below, as an example, the location of T_{off} for the beat is moved from sample 2991 to sample 3011 by clicking and dragging the red dashed line to the right:



2. Edit the values in the '**Beat Selection**' textbox directly:



3. Click on the **−1** and **+1** buttons under the specified fiducial point in the **'Shift Fiducial Points'** section to increase or decrease the specified fiducial point value by 1 sample. Note that to affect the selected beat you must first uncheck the ☐ Median checkbox so that the changes get applied to the selected beat and not the median beat. Further details on how the 'Shift Fiducial Points' buttons work (there is a change in organization and functionality as of version 1.4.0) please see **Chapter 12.9**.

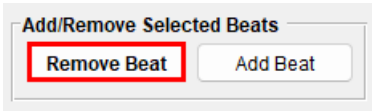


After you use one of these three methods to set the appropriate values for the beat in the '**Beat Selection**' textbox, click **Update Beat** to update the values and recalculate the new median beat and measurements.

Note: The Next and Prev buttons can be used to cycle through the beats in the '**Annotated Beats**' listbox without clicking on each beat individually.

5.7 Removing Beats

To remove a beat, click on the beat to be removed in the '**Annotated Beats**' listbox. The selected beat is shown in the '**Selected Beat Viewer**' and highlighted in green in the '**X-, Y-, Z-, Vector Magnitude Leads**' and '**Median Beat**' sections as noted in the above examples. Once the beat to be removed is selected, click the **Remove Beat** button in the '**Add/Remove Selected Beats**' section:



The selected beat is removed, a new median beat is created and annotated, and measurements are performed on the new median beat.

5.8 Adding Beats

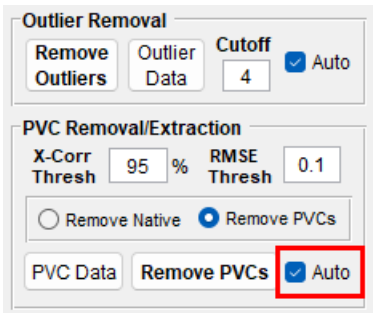
It should be rare to have to add beats manually, but this can be done if needed. Enter the respective fiducial points in the '**Beat Selection**' text box and then click Add Beat in the '**Add/Remove Selected Beats**' section:

The new beat is added, a new median beat is created and annotated, and measurements are performed on the new median beat.

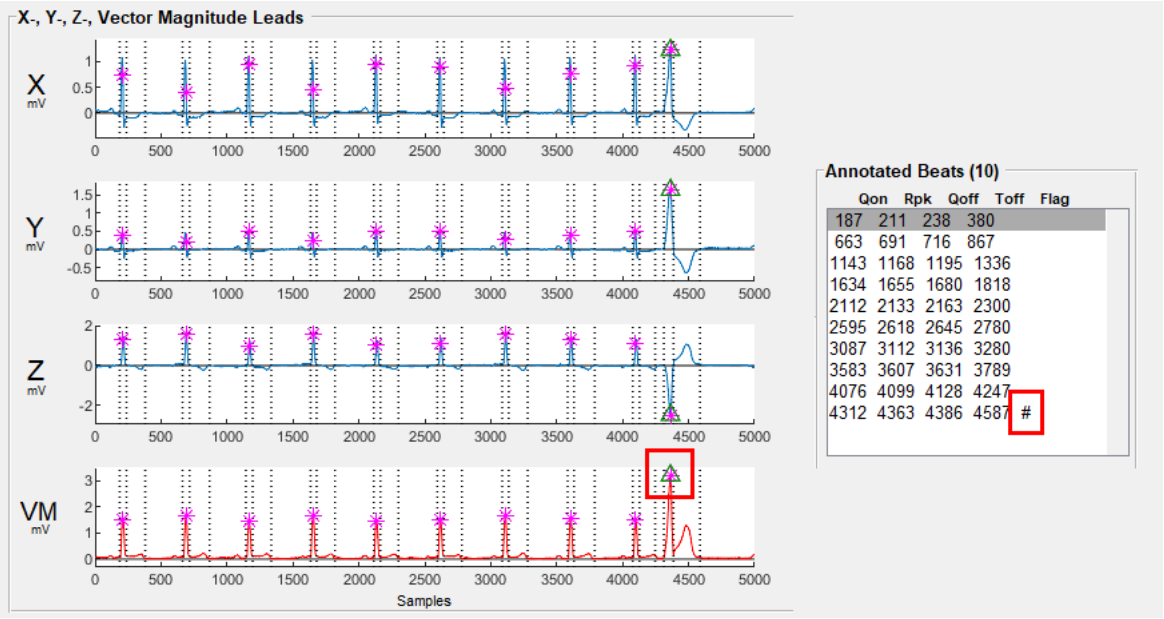
Note: Manually added beats are inserted at the end of the '**Annotated Beats**' list and not in the temporal order in which they appear in the ECG. This has no effect on annotation or calculations.

5.9 Removing PVCs and other Non-Dominant Beats

A discussion of the PVC (or any non-dominant QRST morphology) detection and removal algorithm can be found in **Chapter 15.4** and/or the methods manuscript [1]. Additionally, Example 2 (**Chapter 5.2**) shows an example of PVC removal. If ☑ Auto is checked in the '**PVC Removal/Extraction**' section, PVCs will be automatically detected and removed when the **Calculate** button is clicked. If PVC removal is disabled (☐ Auto), PVCs will be detected but not removed as noted above in Example 2.



After the **Calculate** button is clicked, PVCs are noted in the '**X-, Y-, Z-, Vector Magnitude Leads**' section with a ⚠, and in the '**Annotated Beat**' list with a '#' following the numbered fiducial points.

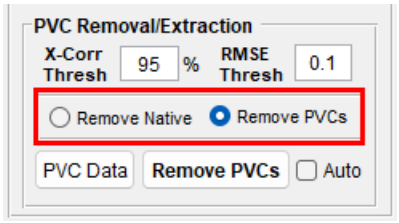


In the figure above, for illustrative purposes, outlier detection (see **Chapter 5.10** and **Chapter 15.5**) was disabled. In many cases, PVCs also get flagged as outliers, and the GUI will note the beat as both a PVC and outlier as shown in Example 2 (**Chapter 5.2**).

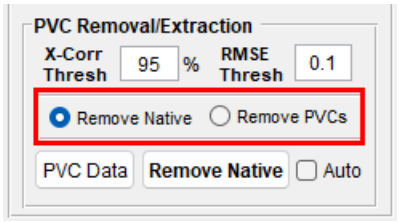
Data about how BRAVEHEART is classifying each beat for PVC detection can be displayed in the GUI by clicking the PVC Data button which displays a figure showing which beats are being detected as PVCs and why. Further details are available in **Chapter 20** and **Figure 41**.

PVCs that have been detected but not removed can be easily removed all at once by clicking the **Remove PVCs** button. Alternatively, PVCs can be manually removed one at a time by selecting each PVC in the '**Annotated Beat**' list and then clicking the **Remove Beat** button as described in the previous examples.

If you wanted to analyze the PVC in the `example2.xml` rather than removing it, use the radio button which allows you to select if 'PVC Removal' will remove or keep PVCs. The default setting is to 'Remove PVCs':



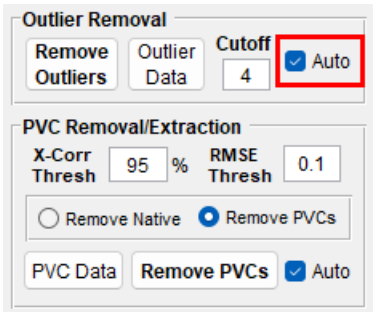
By choosing 'Remove Native' in the radio button, note how the **Remove PVCs** button changes to the **Remove Native** button:



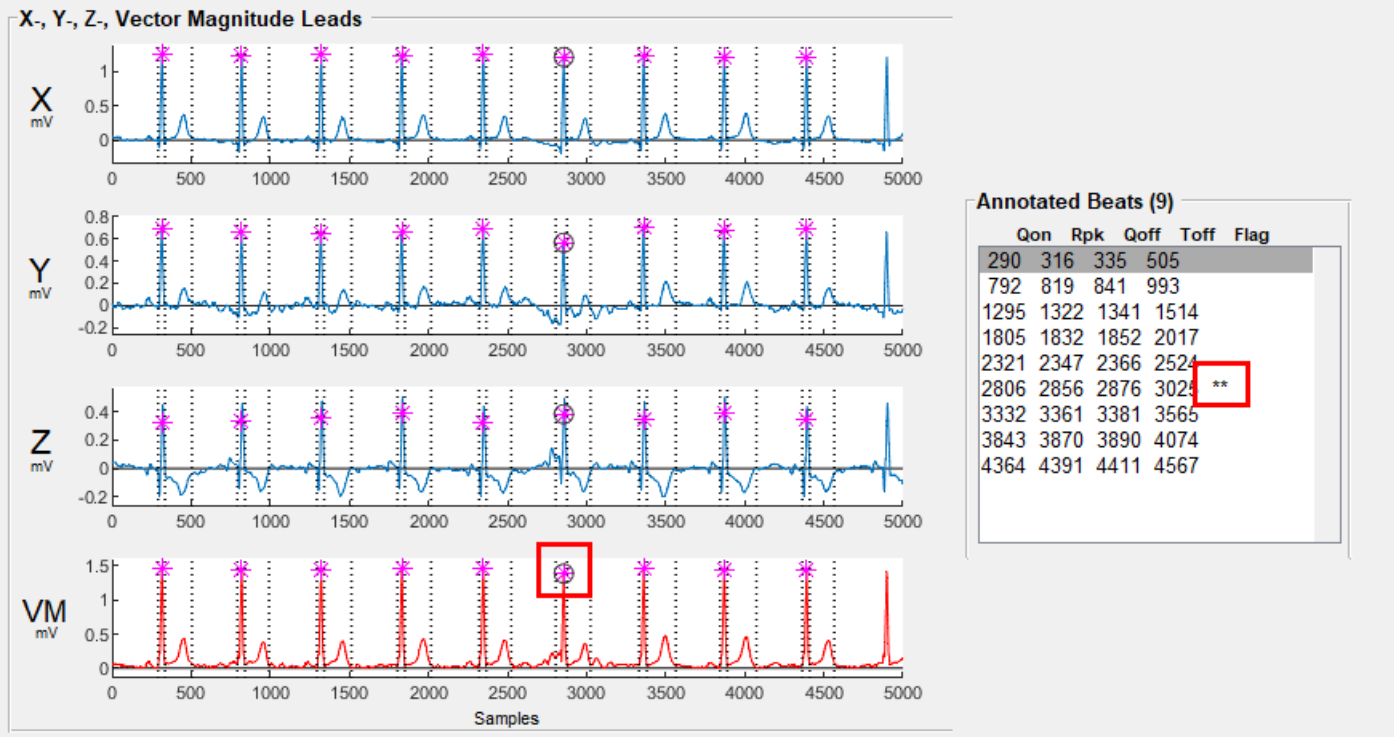
After an ECG is processed with the **Calculate** button, if 'Remove Native' is selected in the GUI radio button, automatic PVC removal and the **Remove Native** button will actually remove the non-PVC beats and keep the PVCs/non-dominant beats for analysis.

5.10 Removing Outlier Beats

A discussion of the outlier detection algorithm can be found in **Chapter 15.5** and/or the methods manuscript [1]. If ☑ Auto is checked in the '**Outlier Removal**' section, outliers will be automatically detected and removed when **Calculate** is pressed. If outlier removal is disabled (☐ Auto), outliers will be detected but not removed.



Outliers are noted in the '**X-, Y-, Z-, Vector Magnitude Leads**' section with a ⊛, and in the '**Annotated Beat**' list with a '**' following the numbered fiducial points:



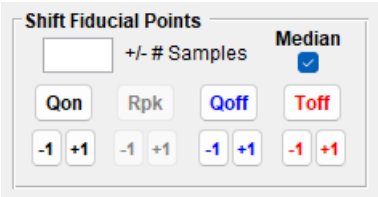
Data about how BRAVEHEART is classifying each beat for outlier detection can be displayed in the GUI by clicking on the Outlier Data button. The QR and RT intervals are used for outlier detection, and beats that are considered as outliers will have their QR and/or RT interval displayed with a red background. The other measures that are displayed, such as the RS, JT, and QT intervals and area under the QRST complex are displayed for the users information but are not used to actually flag a beat as an outlier, and are displayed with a black background. See **Chapter 15.5** and **Figure 42** for further details.

Outliers that have been detected but not removed can be easily removed all at once by clicking the **Remove Outliers** button. Alternatively, outliers can be manually removed one at a time by selecting each beat in the '**Annotated Beat**' list and then clicking the **Remove Beat** button. Note that outliers and PVCs are reassessed after each beat is removed, so it is possible that a beat which is classified as an outlier prior to another beat being removal may no longer be classified as an outlier after the other beat is removed. Additionally, some outliers may also be marked as PVCs, and PVC removal may remove them.

5.11 Manually Adjusting Median Beat Annotations

Median beat fiducial point annotation nominally utilizes a custom neural network (NN) developed specifically for median beat annotation (see the methods manuscript [1] and **Chapter 19** for details). We have demonstrated that the NN has very high accuracy, but rarely annotations may be inaccurate. There is some subjectivity in annotating QRST fiducial points, especially the T_{off}, so we would recommend not manually editing the median beat fiducial points unless the provided points are significantly abnormal to maintain consistency and reproducibility of the results. In cases where the NN provided fiducial points are significantly abnormal, or in niche cases where manual annotation is required, the median beat fiducial points Q_{on}, Q_{off}, and T_{off} can be manually adjusted as described below.

In the '**Shift Fiducial Points**' section below the '**Selected Beat Viewer**', make sure ☑ Median is checked. Enter a number of *samples* (positive numbers shift towards the right/later, and negative numbers shift towards the left/earlier) in the textbox. Then press the button for the fiducial point you want to shift:



For example, to shift the median beat T_{off} later by 3 samples, enter '3' in the textbox and click the Toff button. To shift the median beat Q_{on} earlier by 2 samples enter '-2' in the textbox and click the Qon button. The fiducial points of the median beat are adjusted and calculations are updated.

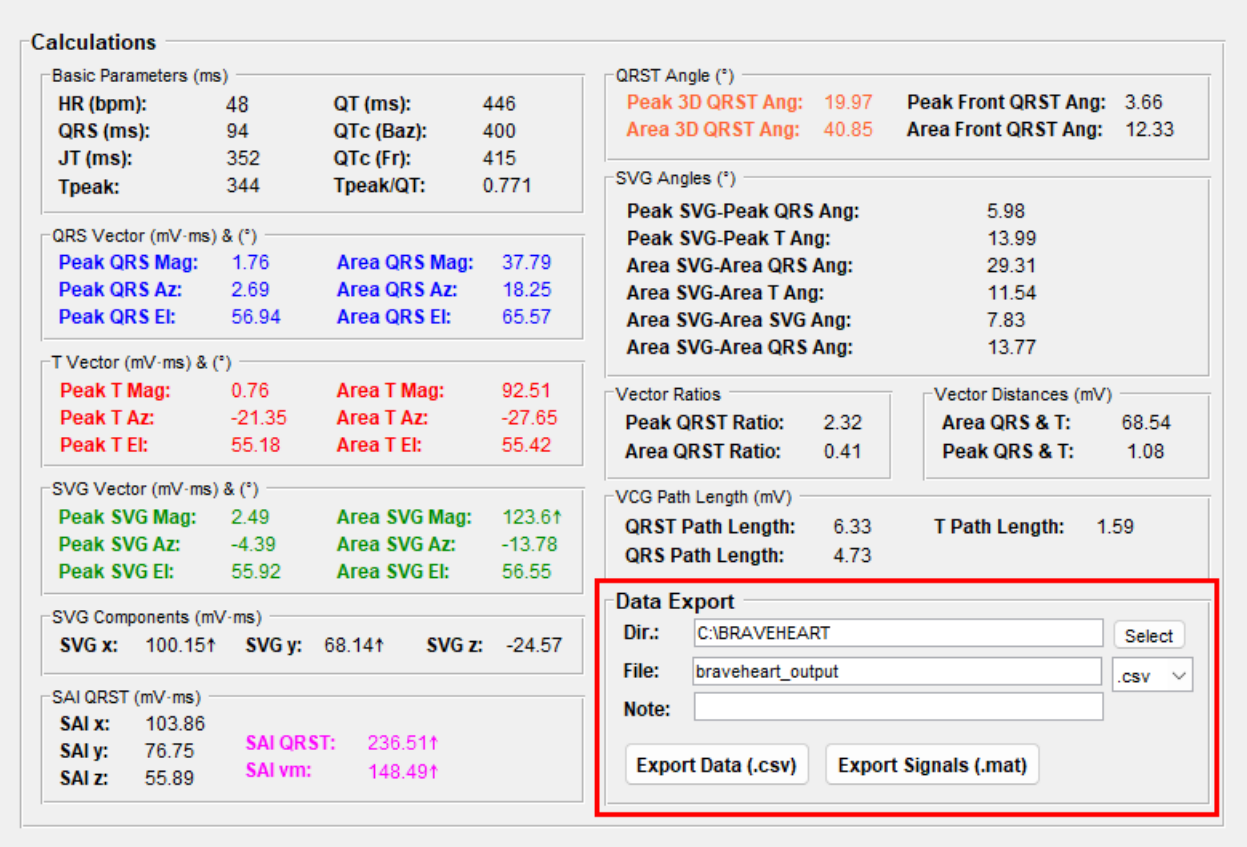
If ☑ Median is checked, clicking the −1 and +1 buttons under the specified fiducial point will adjust the specific fiducial point by ±1 sample.

Note that the R peak of the median beat CANNOT be adjusted manually as this is a fixed fiducial point with minimal ambiguity, and the buttons to adjust the R peak value only are active if ☐ Median is unchecked indicating that any adjustments are on individual beats and not the median beat.

5.12 Exporting Measurements to a Text File

BRAVEHEART can output results to a text file for analysis. After you have processed the ECG and made any manual edits, click the **VCG Calculations** button in the middle of the GUI as shown in Example 1 (**Chapter 5.1**).

This brings up the '**VCG Calculations**' section which displays various VCG measurements and is used for exporting all calculations. The controls for data export are shown in the bottom right:



The default directory used to save files is the same directory from which the ECG was loaded. To change the directory for saving click Select and choose a new directory. The default file name for the saved data file is `braveheart_output`; change this to what you want to name the output file.

Once your file name and directory are set, click **Export Data (.csv)** to export the data to a `.csv` file located in the directory chosen above and with the specified file name. Once the file is successfully exported, **Success!** is displayed in green in the bottom right of the section.

5.13 Exporting Signals

Chapter 5.12 describes how to export calculations into a text file for further analysis. BRAVEHEART can also export these calculation data, and all ECG, VCG, median beat signals, and annotation data, in a MATLAB `.mat` file which contains a data structure with all results/signals. In the '**VCG Calculations**' section, click the **Export Signals (.mat)** button. A file with the same name as the ECG and with extension `.mat` is created the the directory specified for data export.

5.14 Exporting Figures

A variety of figures can be generated and saved by the BRAVEHEART GUI. See **Chapter 20** for additional information.

6 Annotation Parameters (`Annoparams.m`)

`Annoparams.m` contains parameters which control the behavior of BRAVEHEART's ECG/VCG signal processing and annotation algorithms. These parameters can be set globally by editing the `Annoparams.m` file. In cases where BRAVEHEART is being run via executable and `Annoparams.m` cannot be edited, the parameters can be set by editing `Annoparams.csv` which is an external file with name/value pairs separated by commas. When the program is run via executable, calls to pull data from `Annoparams.m` will instead pull data from `Annoparams.csv`. Note that when editing `Annoparams.csv`, unlike when editing `Annoparams.m`, strings should **not** be enclosed in quotes.

When using the GUI it should be rare to have to edit the `Annoparams.m` (or `Annoparams.csv`) file because almost all of the important parameters that would be set in `Annoparams.m` (or `Annoparams.csv`) can be set via dropdowns, checkboxes, or textboxes in the GUI. However, editing `Annoparams.m` (or `Annoparams.csv`) can be used to change the default values when the GUI loads. The correspondence between GUI elements and specific properties of `Annoparams.m` is described in **Chapter 12** and **Chapter 29**.

When using the GUI, values are pulled from `Annoparams.m` (or `Annoparams.csv`) when the GUI is loaded/initialized for the first time and when GUI parameters are reset with the Reset button (see **Chapter 12**). The actual GUI settings at the time an ECG is processed or a calculation is run will override any settings within `Annoparams.m` (or `Annoparams.csv`).

The following sections describe the properties within the `Annoparams.m` class that can be set by the user (either via editing files or the GUI) to change the behavior of BRAVEHEART.

6.1 Annoparams Properties (`Annoparams.m`)

`Annoparams.m` properties are listed here and descriptions of their use follows in **Chapter 6.2**.

```
maxBPM = 150;                    % Sets window for detecting R peaks
pkthresh = 95;                   % Percentile of ECG signal above which a peak can be found
pkfilter = 0;                    % Additional filtering prior to peak detection on/off
lowpass = 1;                     % Low pass wavelet filter on/off
highpass = 1;                    % High pass wavelet filter on/off
wavelet_level_lowpass = 1;       % LPF freq is > samp freq/2^(wavelet_level_lowpass + 1)
wavelet_level_highpass = 10;     % HPF freq is < samp freq/2^(wavelet_level_highpass + 1)
wavelet_name_lowpass = 'sym4';   % Low-pass wavelet (Sym4, Sym5, Sym6, db4, db8, db10, etc)
wavelet_name_highpass = 'db4';   % High-pass wavelet (Sym4, Sym5, Sym6, db4, db8, db10, etc)
baseline_correct_flag = 1;       % Corrects baseline offset

transform_matrix_str = 'Kors';   % Kors or Dower transformation matrix

baseline_flag = 'zero_baseline'; % Zero reference for area calculations
origin_flag = 'zero_origin';     % Origin for VCG plotting

autoMF = 1;                      % Auto estimate QRS width
autoMF_thresh = 20;              % Percent Rpeak threshold if autoMF = true
MF_width = 40;                   % Length of median filter (in ms) used when autoMF = true

QRwidth = 100;                   % Width of QR search window in ms
RSwidth = 100;                   % Width of RS search window in ms
STstart = 100;                   % Distance between Qoff and start of Tend search window in ms
STend = 45;                      % Length of Tend search window as a percent of RR interval

spike_removal = 1;               % Remove pacemaker spikes
pacer_spike_width = 20;          % Max width of pacing spike (in ms)
pacer_mf = 4;                    % Pacer spike detection median filter (in ms)
pacer_thresh = 20;               % Percent peak of pacer spike used for spike removal

cwt_spike_removal = 0            % Remove pacemaker spikes using CWT
interpolate = 0;                 % Remove pacing spikes with interpolation
pacer_zcut = 65;                 % Mod Z score cutoff for detecting pacing spikes
pacer_zpk = 20;                  % Percent peak of mod Z score peaks defining onset/offset of spike
pacer_maxscale = 1;              % Lower limit of frequency used for pacing spike detection
pacer_spike_num = 2;             % Minimum number of leads that have to have pacing spikes detected

align_flag = 'CoV';              % Beat alignment method ('CoV' or 'Rpeak')
cov_mf = 40;                     % width of CoV median filter (in ms)
cov_thresh = 30;                 % CoV median filter threshold %
shiftq = -40;                    % Q window expand when calculating median beat (in ms)
shiftt = 60;                     % T window expand when calculating median beat (in ms)

Tendstr = 'Energy';              % Tend detection method ('Energy', 'Tangent', or 'Baseline')

median_reanno_method = 'NNet';   % 'NNet' for neural network and 'Std' for standard annotations

outlier_removal = 1;             % Remove outliers
modz_cutoff = 4;                 % Cutoff for mod Z-score to flag an outlier (higher less sensitive)
pvc_removal = 1;                 % Remove PVCs
pvcthresh = 0.95;                % Cross correlation threshold for PVC removal
rmse_pvcthresh = 0.1;            % Normalized RMSE threshold for PVC removal
```

```
keep_pvc = 0;                    % Set = 1 if PVC removal removes native QRS instead of PVCs

blanking_window_q = 0;           % Blanking window after QRS onset (in ms) to ignore in speed calcs
blanking_window_t = 20;          % Blanking window after TW onset (in ms) to ignore in TW speed calcs

debug = 0;                       % Debug mode (generates debug annotation figures)
```

6.2 Annoparams Property Descriptions

`property` | *acceptable values*

property description

`maxBPM` | Positive numeric values

Length of the blanking window in beats per min (bpm) centered around each dominant R peak in the VM signal where a second R peak will not be found. This is also the lower limit of the heart rate (HR) that can be detected by the R peak detector; `maxBPM` must be set to a value above the HR on the ECG to avoid missing QRS complexes. The blanking window in ms (w) given as $w = 60000/$ `maxBPM`. The nominal value of `150` bpm corresponds to a 400 ms window centered on the highest peaks detected in the VM signal. Additional information can be found in **Chapter 15.1** and **Figure 24** which illustrates R peak detection using the `pkthresh` and `maxBPM` parameters.

`pkthresh` | Numeric values between `0` and `100`

Cutoff percentile above which an R peak can be detected in the vector magnitude (VM) signal as part of initial R peak detection. This is nominally set to `95`. Higher values may miss some R peaks, while lower values may inappropriately identify peaked T waves or other parts of the ECG signal. However, because the R peak detection algorithm also takes into account a rate related blanking period around each R peak defined by `maxBPM` (see below) where only a single R peak can be detected, as long as the `maxBPM` parameter is set to a reasonable rate, it is rare to over count R peaks unless `pkthresh` and `maxBPM` are both set very low. In cases where the magnitude of PVCs or paced beats in the VM lead is significantly different than the magnitude of the dominant QRST morphology, `pkthresh` may have to be reduced to detect both the dominant and non-dominant R peaks. Additional information can be found in **Chapter 15.1** and **Figure 24** which illustrates R peak detection using the `pkthresh` and `maxBPM` parameters.

`pkfilter` | Integer `0` or `1`

Sets if the VM signal filtering is is activated (`1`) or deactivated (`0`). Filtering is done primarily to remove the T wave prior to R peak detection. The filtered VM beat uses the same values as specified in `pkthresh` and `maxBPM` when finding R peaks. `pkfilter` is nominally set off as it is not necessary in most cases, and it was a new addition in version 1.3.0. Additional information can be found in **Chapter 15.1** and **Figure 24** which illustrates R peak detection using the `pkthresh` and `maxBPM` parameters.

`lowpass` | Integer `0` or `1`

Sets if low-pass wavelet-based filtering is activated (`1`) or deactivated (`0`). Low-pass filtering removes frequencies above a cutoff frequency and helps remove high-frequency noise.

`highpass` | Integer `0` or `1`

Sets if high-pass wavelet-based filtering is activated (`1`) or deactivated (`0`). High-pass filtering removes frequencies below a cutoff frequency and helps remove low-frequency baseline wander and DC offset.

`wavelet_level_lowpass` | Integer values $\geq$ `1`

Corresponds to the low-pass filtering cutoff frequency level. The low-pass cutoff filtering frequency (f_c) is given by $f_c = f_s/(2^{n+1})$, where f_s is the ECG sampling frequency and n is the value of `wavelet_level_lowpass`. As can be seen, the same wavelet level will correspond to a different low-pass filtering cutoff frequency for signals with different sampling frequencies. In general, `wavelet_level_lowpass` should be a low value ≤ 3 for most applications (depending on signal sampling frequency) to avoid removing frequencies contained within the QRST complex (0.5-40 Hz)[3]. The maximum n (maximum value of `wavelet_level_lowpass`) depends on the length of the ECG signal in samples (L) and is given as $n_{\max} = \text{floor}(\log_2 L)$. The BRAVEHEART GUI nominally sets
`wavelet_level_lowpass = 1` for signals sampled at 500 Hz, and
`wavelet_level_lowpass = 2` for signals sampled at 1000 Hz when ECG formats are changed; this corresponds to a nominal low-pass filtering cutoff of 125 Hz. See **Chapter 7** for further details.

`wavelet_level_highpass` | Integer values $\geq$ `1`

Corresponds to the high-pass filtering cutoff frequency level. The high-pass cutoff filtering frequency (f_c) is given by $f_c = f_s/(2^{n+1})$, where f_s is the ECG sampling frequency and n is the value of `wavelet_level_highpass`. As can be seen, the same wavelet level will correspond to a different high-pass filtering cutoff frequency for signals with different

sampling frequencies. In general, `wavelet_level_highpass` should be a high value ≥ 9 to avoid removing frequencies contained within the QRST complex (0.5-40 Hz)[3]. The maximum n maximum value of `wavelet_level_highpass`) depends on the length of the ECG signal in samples (L) and is given as $n_{\max} = \text{floor}(\log_2 L)$. The BRAVEHEART GUI nominally sets `wavelet_level_lowpass = 10` for signals sampled at 500 Hz, and `wavelet_level_lowpass = 11` for signals sampled at 1000 Hz when ECG formats are changed; this corresponds to a nominal high-pass cutoff frequency of 0.24 Hz. See **Chapter 7** for further details.

`wavelet_name_lowpass` | `string` values corresponding to MATLAB wavelet names

Corresponds to the specific wavelet used for low-pass filtering. Nominal settings use `'sym4'`, which corresponds to the Symelets 4 wavelet, and which should be adequate for most ECG signals. A full list of MATLAB wavelets and their associated strings can be found in the MATLAB documentation or by typing `waveinfo` in the MATLAB Command Window.

`wavelet_name_highpass` | `string` values corresponding to MATLAB wavelet names

Corresponds to the specific wavelet used for high-pass filtering. Nominal settings use `'db4'`, which corresponds to the Daubechies 4 wavelet, and which should be adequate for most ECG signals. A full list of MATLAB wavelets and their associated strings can be found in the MATLAB documentation or by typing `waveinfo` in the MATLAB Command Window.

`baseline_correct_flag` | Integer `0` or `1`

Sets if physiological baseline correction is activated (`1`) or deactivated (`0`). This provides additional baseline correction after high-pass filtering to set the isoelectric TP segment (the physiological zero voltage reference) as the approximate zero voltage reference for calculations. Further information can be found in **Chapter 15.3** and **Figure 39**.

`transform_matrix_str` | `'Kors'` or `'Dower'`

Sets the transformation matrix used to transform the ECG into a VCG. The nominal transformation matrix is the Kors transformation matrix [4] which is set by the string `'Kors'`. The inverse Dower transformation [5] can be used with the string `'Dower'`. Additional transformation matrices can be added as shown in **Chapter 9**.

`baseline_flag` | `'zero_baseline'`, `'Tend'`, `'QRS Onset'`, or `'Avg'`

Sets how the zero baseline is defined for area calculations. The nominal setting of `'zero_baseline'` takes 0 mV as the reference line and should be used assuming the signals

are zeroed appropriately using filtering and baseline correction. Other options include `'Tend'` which sets the end of the T wave, `'QRS Onset'` which sets the start of the QRS complex, and `'Avg'` which sets the mean of the QRS onset and T wave end as the zero reference for area calculations, respectively. In general, the nominal setting `'zero_baseline'` should be used (See **Chapter 15.3** and **Figure 38**).

`origin_flag` | `'zero_origin'`, `'Tend'`, or `'Avg'`

Sets the location of the VCG origin. The nominal setting of `'zero_origin'` takes the point (0,0,0) as the origin and should always be used assuming the signals are zeroed appropriately using filtering and baseline correction. `'Tend'` shifts (0,0,0) to the location of the end of the 3D T wave loop, and `'Avg'` shifts (0,0,0) to the point in space equidistant between the onset of the 3D QRS loop and the end of the 3D T wave loop.

`autoMF` | Integer `0` or `1`

Sets if median filtering is used to estimate QRS width as part of heuristic first pass QRS detection/annotation. If `autoMF = 0`, the values of `QRwidth` and `QSwidth` (see below) are used to set the QRS fiducial point detection window. Setting `autoMF = 1` allows BRAVHEART to dynamically adjust the QRS fiducial point search window for each beat individually rather than relying on a fixed window for all beats, and is recommended for most ECGs. In cases where heuristic first pass annotation fails (usually due to atypical QRST morphology), setting `autoMF = 0` and manually adjusting `QRwidth` and `QSwidth` may be useful. See **Chapter 18** for further details.

`autoMF_thresh` | Integers between `0` and `100`

Sets the percentage of each beat's VM QRS peak voltage that is used to estimate each QRS peak width as part of first pass heuristic annotation. If the value of `autoMF_thresh` is set too high, the QRS search window will be too short and the true onset/offset of the QRS complex may be missed. If the value of `autoMF_thresh` is set too low, parts of the T wave may be captured in the QRS search window and inaccuracies in the location of QRS onset/offset may result. The nomimal value of `20` works for the majority of ECGs. Modest increases or decreases in this value may occasionally be needed based on QRS/T wave morphologies. See **Chapter 18** for further details.

`MF_width` | Positive integer values

Sets the width of the median filter (in ms) used when `autoMF = 1`. The nominal value of `40` should be used for most applications and rarely requires change. Note: This property cannot be set via the GUI.

`QRwidth` | Positive integer values

Manually sets the width of the heuristic first pass annotation QRS search window (in ms) **before** the R peak if `autoMF = 0`. This search window is used to locate the QRS onset (Q_{on}). A description of the windows used for heuristic first pass annotation is detailed in **Chapter 18** and **Figure 45**. If `autoMF = 1` this has value no effect on annotation.

`RSwidth` | Positive integer values

Manually sets the width of the heuristic first pass annotation QRS search window (in ms) **after** the R peak if `autoMF = 0`. This search window is used to locate the QRS offset (Q_{off}). A description of the windows used for heuristic first pass annotation is detailed in **Chapter 18** and **Figure 45**. If `autoMF = 1` this value has no effect on annotation.

`STstart` | Positive integer values

Sets the distance (in ms) between the end of the QRS complex and the start of the T wave end search window. A description of the windows used for heuristic first pass annotation is detailed in **Chapter 18** and **Figure 45**.

`STend` | Positive values between `0` and `100`

Sets the width of the T wave end search window starting at `STstart` and going forward by `STend` % of the mean RR interval across the full ECG. `STend` is set as a percentage rather than as absolute value in ms because the QT interval is highly influenced by heart rate, and this reduces the need for frequent adjustment of this parameter. A description of the windows used for heuristic first pass annotation is detailed in **Chapter 18** and **Figure 45**.

`spike_removal` | Integer `0` or `1`

Sets if automatic pacemaker spike/artifact removal is on (`1`) or off (`0`) using the "Width Filter" method. In general, even if pacing is not present, this should be left on as the signals and subsequent annotation will not be affected if pacing is not present. In rare cases, disabling pacemaker spike removal by setting `spike_removal = 0` may resolve errors during ECG loading or annotation and can assist with trouble shooting ECGs that won't load, usually due to very wide or atypical pacing spikes/artifact. Further details can be found in **Chapter 15.2**.

`pacer_spike_width` | Positive integer values

Sets the maximum width of a high-frequency deflection (in ms) in the VM lead that can be defined as a pacing spike when using the "Width Filter" method. The nominal value of `20`

should work in most cases. If there are issues removing very large/wide pacing spikes, adjusting this value may be useful. Further details can be found in **Chapter 15.2**.

`pacer_mf` | Integer `0` or `1`

Sets the width of the median filter (in ms) used when `spike_removal = 1` (when using the "Width Filter" method). The nominal value of `4` should be used for most applications and rarely requires change. Further details can be found in **Chapter 15.2**. Note: This property cannot be set via the GUI.

`pacer_thresh` | Numeric values between `0` and `100`

Sets the percentage of the pacemaker spike amplitude that is used to determine if a high frequency spike is a pacing artifact when using the "Width Filter" method. In most cases, the nominal value of `20` works well, but if there are issues removing very large pacing spikes, adjusting this value may be useful. Further details can be found in **Chapter 15.2**.

`cwt_spike_removal` | Integer `0` or `1`

Sets if automatic pacemaker spike/artifact removal is on (`1`) or off (`0`) using the "CWT Spike Filter" method which was added in version 1.5.0. This method also allows the pacing spikes to be removed from the processed signals rather than just ignored during R peak detection. Although this method is more robust than the "Width Filter" method, it is significantly more computationally expensive, and will slow down ECG processing if enabled. Further details can be found in **Chapter 15.2**.

`interpolate` | Integer `0` or `1`

Sets if the "CWT Spike Filter" method, which is enabled/disabled using the above parameter `cwt_spike_removal` will remove the pacing spikes using hyperbolic cosine interpolation. If this is set to `0` but `cwt_spike_removal = 1`, it functions similarly to when using the legacy method with `spike_removal = 1`. If `cwt_spike_removal = 1` and `interpolation = 1` the spikes are removed from the processed ECG. This can be especially useful for ECGs with very large/wide spikes that cause issues with fiducial point annotation. Further details can be found in **Chapter 15.2**.

`pacer_zcut` | Positive integer values

Sets the modified Z score cutoff for detecting pacing spikes when using the "CWT Spike Filter" method. A pacing spike is detected when the value of the modified Z-score of the difference of the ECG signal exceeds the value of `pacer_zcut`. Lower values are more sensitive but risk false positives, while higher values are more specific but risk false

negatives. The nominal value of `65` works well for most ECGs sampled at 500 and 1000 Hz, but ECGs sampled at higher/lower frequencies may need to adjust the value. Further details can be found in **Chapter 15.2**.

`pacer_zpk` | Positive integer values

Sets the percent of modified Z-score peak height that defines the start/end of a pacing spike when using the "CWT Spike Filter" method. For each detected pacing spike, the onset and offset are defined as `pacer_zpk` % of the maximum value of each spike. The nominal value of `20` works well for most ECGs sampled at 500 and 1000 Hz with typical pacing spike amplitude and width. For very large and wide pacing spikes the value may need to be decreased significantly (extremely large/wide spikes may need values of `1`). ECGs with low sampling rates $<$ 500 Hz may need larger values to avoid including part of the QRS complex. Further details can be found in **Chapter 15.2**.

`pacer_maxscale` | Positive integer values

Sets the maximum scale that is analyzed for pacing spikes after CWT is performed. When using the CWT Method, the nominal value of `1` only includes the 1st scale. A value of `2` would include scales 1-2 etc. The exact pseudo-frequences that the CWT scales correspond to will vary based on signal sampling frequency and other factors. Since pacing spikes should be in the highest frequency content of the ECG, this should rarely have to be changed from its nominal value of `1`. Further details can be found in **Chapter 15.2**.

`pacer_spike_num` | Integer values between 1 and 12

Sets the number of leads in which a pacing spike must be detected to trigger global detection of pacing spikes when using the CWT method. This is designed to avoid unnecessary detection and interpolation due to noise isolated in single lead (or low number of leads). It is important to note that the algorithm will require at least 1 spike detected in `pacer_spike_num` out of 12 leads to process the ECG as a paced ECG. The leads, however, do NOT have to have the exact same spike locations. Further details can be found in **Chapter 15.2**.

`align_flag` | `'CoV'` or `'Rpeak'`

Sets how individual beats are aligned to create a median beat using either the filtered center of voltage (`'CoV'`) or R wave peak (`'Rpeak'`), respectively. The nominal value of `'CoV'` generally performs better than `'Rpeak'` as it is less influenced by noise or by variations in QRS morphology such as are sometimes seen in bundle branch blocks, and should be used for most applications. If median beat alignment is poor when using `'CoV'`, trying `'Rpeak'`

may possibly improve beat alignment.

`cov_mf` | Positive integer values

Sets the width of the median filter (in ms) used to determine the center of voltage. In general this should remain at the nominal value of `40` and rarely requires change. Note: This property cannot be set via the GUI and rarely requires editing.

`cov_thresh` | Numeric values between `0` and `100`

Sets the percentage of the VM QRS peak used for aligning QRST complexes via center of voltage. Note: This property cannot be set via the GUI and rarely requires editing.

`shiftq` | Negative integer values

Sets the value in ms by which the window around the median beat is expanded prior to the onset of the QRS complex. This ensures that the location of the QRS onset is not cut short. The nominal value of `-40` should be used for almost all applications. Making this value more negative will increase the amount of signal that is included in the median beat before the onset of the QRS complex, and making this value less negative will decrease the amount of signal that is included in the median beat before the onset of the QRS complex. Note: This property cannot be set via the GUI.

`shiftt` | Poitive integer values

Sets the value in ms by which the window around the median beat is expanded after the end of the T wave. This ensures that the location of the T wave end is not cut short. The nominal value of `60` should be used for almost all applications. Making this value larger will increase the amount of signal that is included in the median beat after the end of the T wave, while making this value smaller will decrease the amount of signal that is included in the median beat after the end of the T wave. Note: This property cannot be set via the GUI.

`Tendstr` | `'Energy'`, `'Tangent'`, or `'Baseline'`

Sets the method for T wave end detection (T_{off}) when using the heuristic T wave end detector. If set to `'Energy'` it utilizes a validated method that assesses T wave energy towards the end of the T wave that is better able to deal with low amplitude or abnormal T waves than other methods [6]. Setting `Tendstr` to `'Tangent'` utilizes the tangent method, and setting as `'Baseline'` uses baseline crossing. In general, the best results are obtained with `'Energy'`. See **Chapter 18** for further details.

`median_reanno_method` | `'NNet'` or `'Std'`

Once a median beat is created, median beat fiducial points are obtained by annotating the median beat QRST complex. `median_reanno_method` sets how median beats are annotated, using either the standard hieuristic annotater (`'Std'`) or a custom neural network (`'NNet'`). In almost all cases the nominal value of `'NNet'` should be used as it is the most accurate way to annotate median beats. In rare cases where the neural network fails to accurately annotate a median beat, switching to `'Std'` may work, although usually ECGs that fail neural network annotation are ECGs with significant noise and artifact that are difficult to annotate regardless of method. Details of neural network development and annotation performance are available in the methods manuscript [1] and **Chapter 19**.

`outlier_removal` | Integer `0` or `1`

Sets if automatic outlier beat removal is activated (`1`) or deactivated (`0`). When using the GUI, if `outlier_removal = 0` outlier beats will be identified and marked but not removed. See **Chapter 15.5** for additional details.

`modz_cutoff` | Integer `0` or `1`

Sets the threshold for the modified Z-score that is used to define a beat as an "outlier". If the modified Z-score for the QR and/or RT interval of an individual beat is above the `modz_cutoff` value, the beat is flagged as an outlier. Higher values may be needed when there are few beats; if there are > 15 beats a cut off value of `3.5` is reasonable, although if there are < 10 beats a cut off value of approximately `4` may be needed [7]. See **Chapter 15.5** for additional details.

`pvc_removal` | Integer `0` or `1`

Sets if automatic PVC/non-dominant beat removal is activated (`1`) or deactivated (`0`). When using the BRAVEHEART GUI, if `pvc_removal = 0` PVCs/non-dominant beats will be identified and marked but not removed. The `keep_pvc` parameter (see below) controls whether the dominant or non-dominant beats are kept for analysis. See **Chapter 15.4** for further details.

`pvcthresh` | Numeric values

Sets the cutoff (as a percentage) for normalized cross correlation (NCC) as used for PVC/non-dominant beat detection. The nominal value of `95` works well for most applications. The contribution of NCC to PVC/non-dominant beat detection can be disabled by setting `pvcthresh = 0`. See **Chapter 15.4** for further details.

`rmse_pvcthresh` | Numeric values

Sets the cutoff for root-mean square error (RMSE) as used for PVC/non-dominant beat detection. The nominal value of `0.1` works well for most applications. The contribution of RMSE to PVC/non-dominant beat detection can be disabled by setting `rmse_pvcthresh = -1`. See **Chapter 15.4** for further details.

`keep_pvc` | Integer `0` or `1`

Sets if PVCs/non-dominant beats are removed (`0`) or if dominant beats are removed to allow analysis of the PVCs/non-dominant beats (`1`).

`blanking_window_q` | Positive integers

Sets the window (in milliseconds) after QRS onset which is not included in calculations of VCG loop speed (See **Chapter 12.7**). In general this can be left at the nominal value of `0`, but in certain paced ECGs where there is some contamination of the start of the QRS complex with a high amplitude/frequency pacing spike, increasing this value may be useful. In certain situations it may be desirable to also look at speed at specific parts of the QRST complex, and increasing the blanking window can also assist with this by excluding the signal contained within the blanking window from any speed calculations. Blanking windows have no effect on speed-time integral calculations (see **Chapter 24.3**).

`blanking_window_t` | Positive integers

Sets the window (in milliseconds) after T wave onset which is not included in calculations of T wave VCG loop speed (See **Chapter 12.7**). In general this can be left at the nominal value of `20` to avoid calculating the incorrect location/time of the maximum T wave speed due to small errors in the annotation of QRS offset (such that the first few samples of the T wave are actually the last few samples of the QRS complex). This window does NOT affect measurements of the maximum QRST speed or its location, and only applies to measurements of T wave speed. Blanking windows have no effect on speed-time integral calculations (see **Chapter 24.3**).

`debug` | Integer `0` or `1`

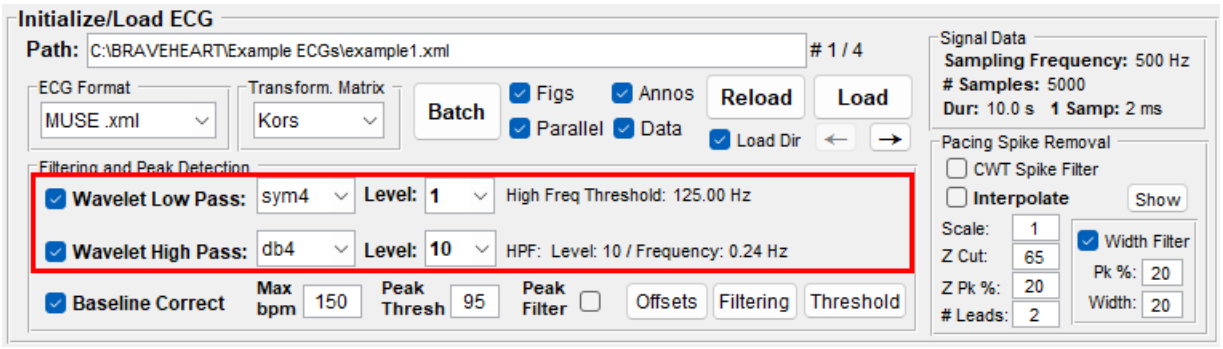
Sets if BRAVEHEART generates debug figures during ECG signal annotation. This can be useful for troubleshooting. Further details can be found in **Chapter 31.8**, **Figure 45**, and **Figure 70**.

---

7 Adjusting Denoising/Filtering Settings

Adjusting signal denoising/filtering appropriately is important for the overall quality of ECG analysis. BRAVEHEART utilizes wavelet based signal denoising/filtering to remove high-frequency artifact and low-frequency baseline wander. Wavelet based denoising/filtering decomposes the signal, removes either high-frequency or low-frequency noise/wander, and then reconstructs the noise free signal. This process is different than standard linear filtering, and helps preserve the underlying features of the ECG with minimal distortion while removing the unwanted noise/wander. The actual denoising process is explained in more detail in our methods manuscript [1], and more information on wavelet decomposition and denoising can be found in other references [8].

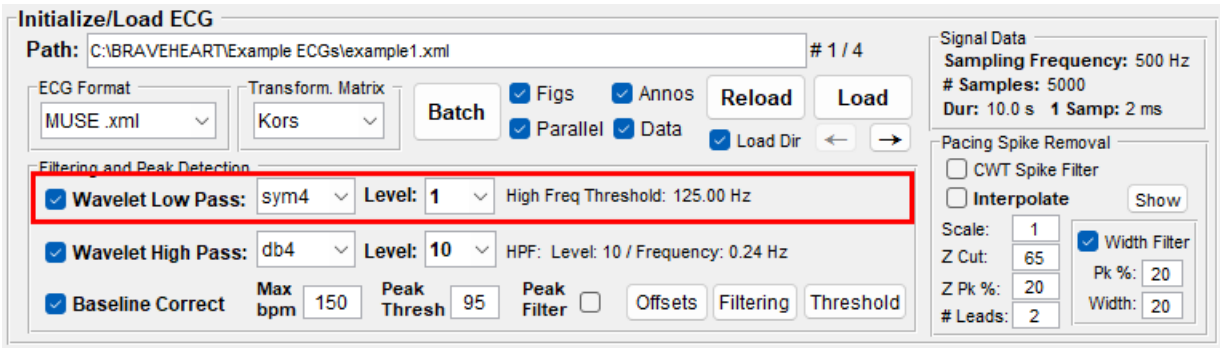
The denoising/filtering settings are controlled in the '**Filtering and Peak Detection**' section of the GUI, but the same principles for adjusting these settings apply when using the command line version of BRAVEHEART via `Annoparams.m` (see **Chapter 6** for additional details).



Additional details on controlling denoising/filtering via the GUI can be found in **Chapter 12.2**.

7.1 High-frequency Denoising (Low-pass Filtering)

High-frequency noise removal is enabled/disabled with the checkbox for ☑ Wavelet Low Pass :



To the right of this checkbox there is a dropdown to choose a mother wavelet (see **Chapter 6** information on `wavelet_name_lowpass`). In the vast majority of cases this can be left alone at the default value of 'Sym4', but this can be changed if needed. The dropdown to the right with selectable integer values sets the wavelet decomposition level that controls the minimum frequency of noise that is removed (see **Chapter 6** information on `wavelet_level_lowpass`):

The Low-pass filtering frequency cutoff (f_c) removes frequencies **above** a cutoff given by the equation $f_c = f_s/2^{n+1}$, where f_s is the ECG sampling frequency and n is a positive, non-zero integer. For example, if the ECG is sampled at 500 Hz, a level (n) of 2 corresponds to a frequency cutoff of $f_c = 500/2^3 = 62.5$ Hz. Therefore, if the level is set $= 2$, the low-pass filtering will remove frequencies higher than 62.5 Hz. Similarly, level 1 with an ECG sampled at 500 Hz corresponds to a frequency cutoff of $f_c = 500/2^2 = 125$ Hz, and level 2 with an ECG sampled at 1000 Hz also corresponds to a frequency cutoff of $f_c = 1000/2^3 = 125$ Hz. A table of wavelet decomposition levels and filtering frequency cutoffs is shown here:

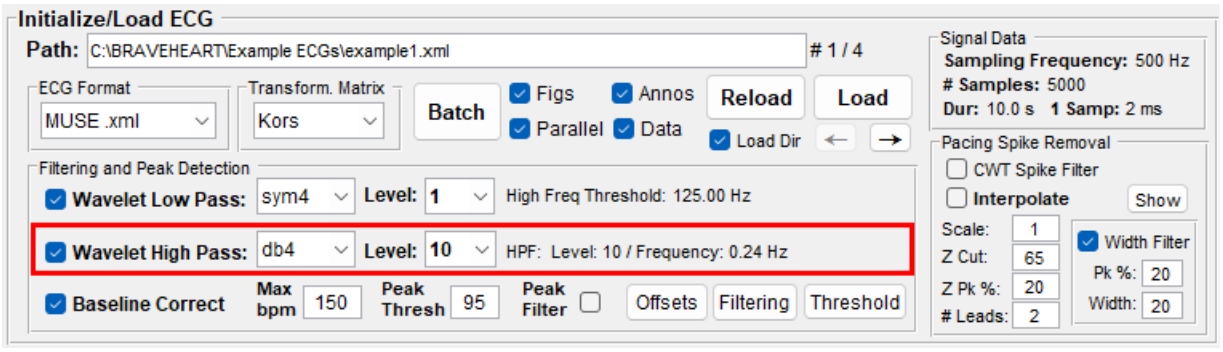
Table 1: Denoising/filtering frequency cutoffs based on wavelet decomposition level (n) and ECG sampling frequency of 500, 1000, and 2000 Hz.

Level (n)	**500 Hz**	**1000 Hz**	**2000 Hz**
1	125.00	250.00	500.00
2	62.50	125.00	250.00
3	31.25	62.50	125.00
4	15.63	31.25	62.50
5	7.81	15.63	31.25
6	3.91	7.81	15.63
7	1.95	3.91	7.81
8	0.98	1.95	3.91
9	0.49	0.98	1.95
10	0.24	0.49	0.98
11	0.12	0.24	0.49
12	0.061	0.12	0.24
13	0.031	0.061	0.12
14	0.015	0.031	0.061

The BRAVEHEART GUI low-pass filtering/denoising defaults to level 1 for ECGs sampled at 500 Hz, and level 2 for ECGs sampled at 1000 Hz, corresponding to a low-pass cutoff of 125 Hz (note the wavelet level in the example figures is set to level 2, which corresponds to a frequency cutoff of 62.5 Hz, not the nominal value of level 1). For ECGs of other sampling frequencies, the level defaults to 1. The Command line version of BRAVEHEART does **not** automatically adjust the wavelet levels based on ECG frequency, and the correct level (not frequency!) must be set in `Annoparams.m` (see **Chapter 6**).

7.2 Low-frequency Denoising (High-pass Filtering) for Baseline Wander Removal

Low-frequency baseline wander removal is enabled/disabled with the checkbox for ☑ Wavelet High Pass :



To the right of this checkbox there is a dropdown to choose a mother wavelet (see **Chapter 6** information on `wavelet_name_highpass`). In the vast majority of cases this can be left alone at the default value of 'db4', but this can be changed if needed, and sometimes mother wavelets with a higher number of vanishing moments (such as db10) may more effectively remove baseline wander. The drop down to the right with selectable integer values sets the wavelet decomposition level that controls the maximum frequency of baseline wander that is removed (see **Chapter 6** information on `wavelet_level_highpass`):

Similar to low-pass filtering, the high-pass filtering frequency cutoff (f_c) removes frequencies **below** a cutoff given by the equation $f_c = f_s/2^{n+1}$, where f_s is the ECG sampling frequency and n is a positive, non-zero integer. Compared to the values of n used for low-pass denoising/filtering which are usually going to be low numbers between 1 and 3 depending on the ECG sampling frequency, to remove low-frequency baseline wander, we have to use higher values of n, usually > 9, depending on ECG sampling frequency (See **Table 1** above). The maximum wavelet decomposition level depends on the length of the signal in samples (L) and is given as $n_{\max} = \text{floor}(\log_2 L)$; for a 10 second ECG sampled at 500 Hz, $n_{\max} = 12$. If you choose an n above $n_{\max}$ the software will give an error and the ECG will not process.

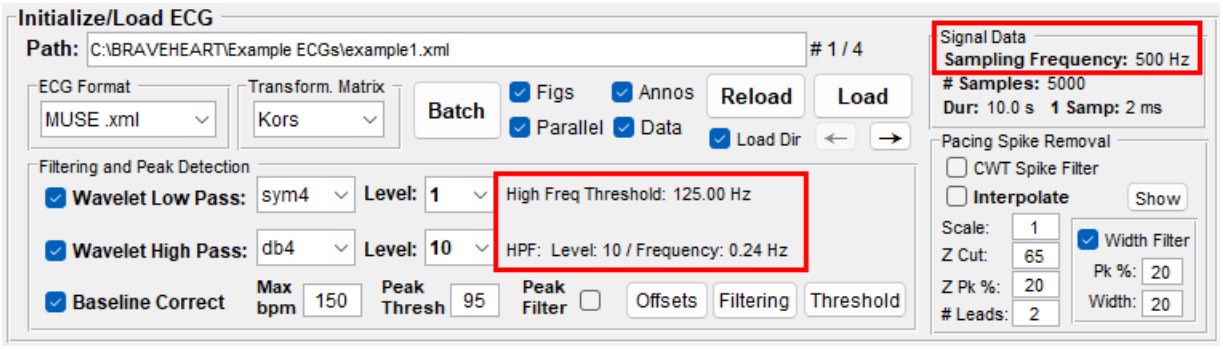
For example, if the ECG is sampled at 500 Hz, a level of 10 corresponds for a frequency cutoff of $f_c = 500/2^{11} = 0.244$ Hz, and a level of 9 corresponds for a frequency cutoff of $f_c = 500/2^{10} = 0.488$ Hz. If the ECG sampling frequency is 1000 Hz, a level of 11 corresponds for a frequency cutoff of $f_c = 1000/2^{12} = 0.244$ Hz and a level of 10 corresponds for a frequency cutoff of $f_c = 1000/2^{11} = 0.488$ Hz (see **Table 1** above).

Importantly, high-pass filtering can distort the ST segment and T wave due to the fact that these parts of the ECG signal can contain low frequency components with the same frequencies as the baseline wander. In general, you should keep the high-pass filtering frequency as low as possible, and use of values > 0.67 Hz is not recommended [9]. We have found that a value of 0.24 Hz works well for most applications, and that this could be increased to ~ 0.5 Hz if needed

without significant distortion. When using the GUI, when an ECG is loaded, the high-pass wavelet level is automatically chosen as the highest level with a cutoff frequency < 0.25 Hz based on the frequency noted in the external file `ecg_formats.csv` (see **Chapter 8.3**). If other levels/frequencies are needed, the levels can be changed and the ECG can be reloaded.

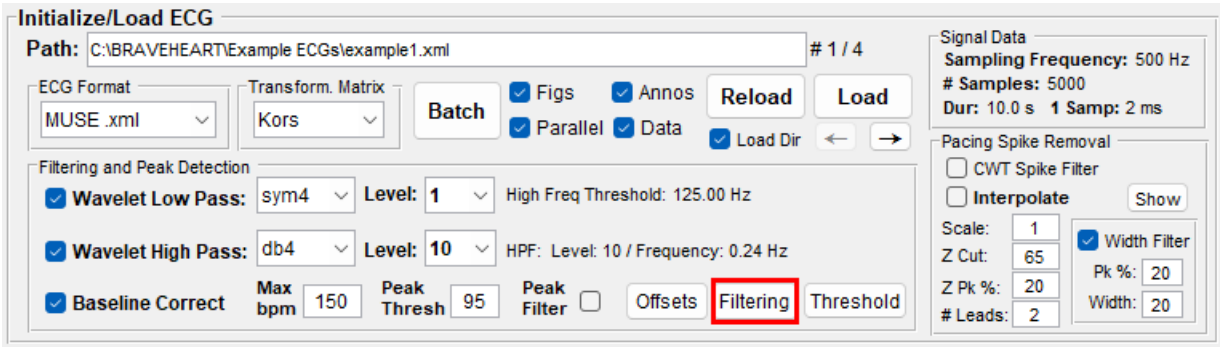
7.3 Denoising/Filtering Frequencies and Visualizing Results

After the the ECG is loaded, text to the right of the wavelet level selection dropdowns will show the filtering frequency that corresponds to the ECG sampling frequency and chosen wavelet decomposition level:



This section will also display an error if you choose a wavelet level greater than the maximum. Note that in this example, the ECG is sampled at 500 Hz (see 'Signal Data' in the upper right), the low-pass wavelet level of 2 corresponds to a frequency cutoff of 62.5 Hz, and the high-pass wavelet level of 10 corresponds to a frequency cutoff of 0.24 Hz. The wavelet based denoising/filtering will therefore remove high-frequency noise above 62.5 Hz and will remove baseline wander below 0.24 Hz. In reality, these frequency cutoffs are not absolute, and there will be some leakage of frequencies around these cutoff points. Additionally, a significant benefit of wavelet based denoising over standard linear filtering is that wavelet based denoising is able to remove noise while preserving the original signal even if the signal shares frequencies that are to be removed with denoising. Further details can be found in our methods manuscript [1] and other wavelet references [8].

The Filtering button displays the original signals and the filtered and baseline corrected signals so that you can visualize the denoising process:



Further details can be found in **Chapter 20.13** and **Chapter 12.2**.

7.4 Denoising/Filtering Using `braveheart_batch.m`

The command line version of BRAVEHEART `braveheart_batch.m` does **NOT** automatically adjust any denoising/filtering settings based on ECG frequency as occurs when using the BRAVEHEART GUI. To set the filtering settings, set the relevant values in `Annoparams.m` before running `braveheart_batch.m`:

```
...
lowpass = 1;                    % Low pass wavelet filter on/off
highpass = 1;                   % High pass wavelet filter on/off
wavelet_level_lowpass = 1;      % LPF freq is > samp freq/2^(wavelet_level_lowpass + 1)
wavelet_level_highpass = 10;    % HPF freq is < samp freq/2^(wavelet_level_highpass + 1)
wavelet_name_lowpass = 'sym4';  % Low-pass wavelet (Sym4, Sym5, Sym6, db4, db8, db10, etc)
wavelet_name_highpass = 'db4';  % High-pass wavelet (Sym4, Sym5, Sym6, db4, db8, db10, etc)
...
```

Note that you will have to do your own calculation (see **Table 1**) to determine what frequency cutoff corresponds to the values of `wavelet_level_lowpass` and `wavelet_level_highpass` based on the ECG sampling frequency.

8 Adding New ECG Formats

BRAVEHEART can read a variety of ECG data formats (see **Chapter 23**). However, if you are using an ECG file format which cannot be read by BRAVEHEART currently, you will have to supply your own load module and modify the relevant part of the `ECG12` constructor in the file `ECG12.m`. Fortunately, adding new ECG formats to BRAVEHEART is straightforward, and only 2 files (`ECG12.m` and `ecg_format.csv`) need to be edited as described in the following sections:

8.1 The `ECG12` Object

12-lead ECG data is stored in an `ECG12` object which contains the following 14 properties:

- `hz` – the ECG sampling frequency in Hz
- `units` – an optional `string` describing ECG voltage units (usually `'mV'`)
- `I` – voltage data for ECG lead I
- `II` – voltage data for ECG lead II
- ... `III, avR, avL, avF, V1, V2, V3, V4, V5, V6` – voltage data for these leads

Note that only 2 of the 6 frontal leads (`I, II, III, avR, avL, aVF`) contain unique information.

The data within an `ECG12` object is accessed by querying its properties as if it were a normal data structure in MATLAB; for example, if you have an `ECG12` object named `ecg`, lead II would be accessed as `ecg.II`, and the sampling frequency would be accessed as `ecg.hz`.

8.2 Editing `ECG12.m`

We will start by adding a new format (called "New Format") to BRAVEHEART. At line 66 of `ECG12.m` you will see a `switch` statement which constructs `ECG12` objects based on a source format string (`format`), such as `'bidmc_format'` or `'muse_xml'`, and a function to load the ECG data from a file:

```
% ADD NEW ECG FORMATS TO THIS SWITCH STATEMENT
switch format

 case 'bidmc_format'
   obj.hz=500;
   unitspermv=200;
   [obj.I, obj.II, obj.III, obj.avR, obj.avF, obj.avL, ...
     obj.V1, obj.V2, obj.V3, obj.V4, obj.V5, obj.V6] = load_ecg(filename, unitspermv, format);

 case 'muse_xml'
   [obj.hz, obj.I, obj.II, ...
     obj.V1, obj.V2, obj.V3, obj.V4, obj.V5, obj.V6] = load_musexml(filename);
   obj.III = -obj.I + obj.II;
   obj.avF = obj.II - 0.5*obj.I;
   obj.avR = -0.5*obj.I - 0.5*obj.II;
   obj.avL = obj.I - 0.5*obj.II;

   ...

end
```

We will add a new `case` statement with a function that generates all of the parts of an `ECG12` object (the `units` property is optional).

For example, if you have a function `load_new_format.m` which loads an ECG in `'new_format'` format from a `.txt` file (input variable name `filename`), the outputs of the function should include the frequency (`obj.hz`) and the 12 ECG leads (`obj.<lead>`). Note that you can supply all 12 leads as the output of the load function (see `'bidmc_format'` in the code block above), or you can supply the 8 independent leads in the load function and then reconstruct the 4 additional frontal plane leads in `ECG12.m` itself (see `'muse_xml'` in the code block above).

After this addition, you will have something that looks like this

```
% ADD NEW ECG FORMATS TO THIS SWITCH STATEMENT
switch format

 case 'bidmc_format'
   obj.hz=500;
   unitspermv=200;
   [obj.I, obj.II, obj.III, obj.avR, obj.avF, obj.avL, ...
     obj.V1, obj.V2, obj.V3, obj.V4, obj.V5, obj.V6] = load_ecg(filename, unitspermv, format);

 case 'muse_xml'
   [obj.hz, obj.I, obj.II, ...
     obj.V1, obj.V2, obj.V3, obj.V4, obj.V5, obj.V6] = load_musexml(filename);
   obj.III = -obj.I + obj.II;
   obj.avF = obj.II - 0.5*obj.I;
   obj.avR = -0.5*obj.I - 0.5*obj.II;
   obj.avL = obj.I - 0.5*obj.II;

 case 'new_format'
   [obj.hz, obj.I, obj.II, obj.III, obj.avR, obj.avF, obj.avL, ...
     obj.V1, obj.V2, obj.V3, obj.V4, obj.V5, obj.V6, obj.avL] = load_new_format(filename);

   ...

end
```

Editing of `ECG12.m` is now complete.

8.3 Editing `ecg_formats.csv`

In addition to the ECG constructor, you will need to modify `ecg_formats.csv` to use your new format. `ecg_formats.csv` is an external, comma delimited, editable file that links the ECG `format` with associated file extensions and labels. `ecg_formats.csv` is also the way that new ECG formats are added to the BRAVEHEART GUI without the hassle of needing to edit the GUI figure.

```
MUSE .xml,          muse_xml,         .xml,  500
BIDMC .txt,         bidmc_format,     .txt,  500
Prucka .txt,        prucka_format,    .txt,  997
Philips .xml,       philips_xml,      .xml,  500
ISHNE .ecg,         ISHNE,            .ecg,  1000
GE .mrq,            mrq_ascii,        .mrq,  500
DICOM .dcm,         DICOM,            .dcm,  500
HL7 .xml,           hl7_xml,          .xml,  500
Unformatted .txt,   unformatted,      .txt,  500
Cardiosoft .xml,    cardiosoft_xml,   .xml,  500
SCP-ECG .scp,       scp_ecg,          .scp,  500
```

Note that the contents of your `ecg_formats.csv` file may be slightly different than as reproduced here as more formats are added over time, and some rows were omitted in the interest of saving space for this example.

The first column in `ecg_formats.csv` is the text that shows up in the GUI when selecting the ECG format (New Format .txt). This can be a string with punctuation or spaces. Column 2 is the ECG format string (`format`) which was added to the `switch` statement above (`'new_format'`), column 3 is the new ECG format file extension (`.txt`), and column 4 is the sampling frequency of the ECG signals which is used by the GUI to select the nominal wavelet levels for filtering. The actual frequency used in calculations is determined when the ECG is actually loaded from the file (see section **Chapter 8.1** describing the `ECG12` class which includes sampling frequency (`hz`) as a property), and the frequency in column 4 of `ecg_formats.csv` is only used to update the nominal filtering settings in the GUI when a different format is chosen from the 'ECG format' dropdown. The frequency in column 4 has no effect when BRAVEHEART is being run from `braveheart_batch.m` where all filtering is set in `Annoparams.m` (or `Annoparams.csv` if the program is compiled).

For example, if ECGs of the format (`'new_format'`) are sampled at 1000 Hz, but the `ecg_formats.csv` file column 4 lists the sampling frequency for this format at 500 Hz, the low-pass filtering wavelet level will be set to 1 (nominal for 500 Hz) instead of 2 (nominal for 1000 Hz) when you choose the (`'new_format'`) format in the GUI dropdown (see **Chapter 7**). The difference in wavelet filtering settings will slightly change the filtered ECG/VCG signals and therefore slightly change subsequent calculations, but all calculations will use the correct sampling frequency of 1000 Hz, because the correct sampling frequency would be pulled from the `ECG12` object `hz` property when the file is loaded (note that some file formats have the sampling frequency hard coded when they are loaded by `ECG12` if that information is not part of the ECG data file). As the command line version of BRAVEHEART does not utilize any part of the GUI, the frequencies in column 4 of `ecg_formats.csv` are ignored when using `braveheart_batch.m`.

To add the `'new_format'` ECG format as shown above to `ecg_formats.csv`, you would add a new line to `ecg_formats.csv` including a name for the GUI, format name, format file extension, and sampling frequency as shown here:

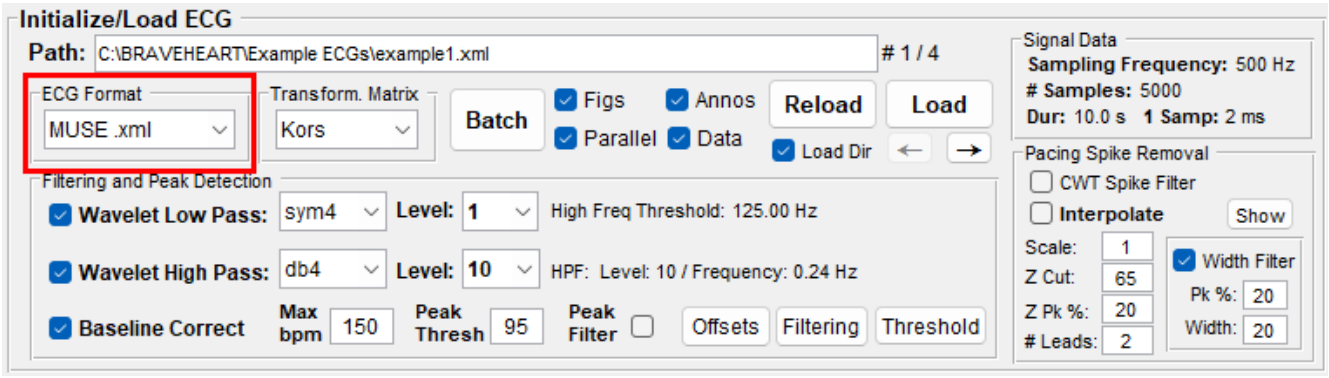
```
MUSE .xml,          muse_xml,         .xml,  500
BIDMC .txt,         bidmc_format,     .txt,  500
Prucka .txt,        prucka_format,    .txt,  997
Philips .xml,       philips_xml,      .xml,  500
ISHNE .ecg,         ISHNE,            .ecg,  1000
GE .mrq,            mrq_ascii,        .mrq,  500
DICOM .dcm,         DICOM,            .dcm,  500
HL7 .xml,           hl7_xml,          .xml,  500
Unformatted .txt,   unformatted,      .txt,  500
Cardiosoft .xml,    cardiosoft_xml,   .xml,  500
SCP-ECG .scp,       scp_ecg,          .scp,  500
New Format .txt,    new_format,       .txt,  500
```

This will now link the `format` `'new_format'` to the file extension `.txt`. When using the GUI, the string "**New Format .txt**" will show up in the ECG format dropdown, and selecting this format in will automatically choose nominal filtering settings for an ECG sampled at 500 Hz.

8.4 Using Newly Added ECG File Formats

If you are using the command line version of BRAVHEART (see **Chapter 13**), simply use your new `format` string (in the above example `'new_format'`) in the header of `BRAVEHEART_batch.m` or `batch_settings.csv` if you are using the compiled version (remember that when editing `batch_settings.csv` that strings should NOT be surrounded by quotes).

If you are using the GUI version of BRAVEHEART, simply select the new format from the 'ECG format' dropdown (see GUI item 1.1 in **Chapter 12**).



The default ECG format selected when loading the GUI is the first row in `ecg_formats.csv`.

If your ECG format includes data such as age and sex, you can add the relevant data to `xml_demographics.m` to automatically pull this data into the GUI for use when viewing normal ranges of various VCG parameters (see **Chapter 22**).

9 Adding New Transformation Matrices

BRAVEHEART includes the Kors [4] and inverse Dower [5] transformation matrices to convert the 12-lead ECG into a VCG using 8 independent ECG leads. In certain situations, alternative transformation matrices may be needed. This section will illustrate how it is straightforward to add a new transformation matrix to BRAVEHEART. For illustrative purposes we will add a new transformation matrix called the "NewMatrix" transformation.

9.1 Create a Function for The New Transformation Matrix

We will start by creating a function `newmatrix()` in a file `newmatrix.m` which simply contains the coefficients for the new transformation matrix. This should be a 3×8 matrix with rows corresponding to X, Y, and Z, and columns ordered as V1, V2, V3, V4, V5, V6, I, and II:

```
function M = newmatrix()
        % Note: These are random numbers
        %       V1     V2     V3     V4     V5    V6    I      II
        M = [ -0.23   0.15  -0.02   0.13   0.10  0.62  0.44  -0.09 ;  % X
               0.10  -0.12  -0.12   0.10  -0.11  0.13 -0.11   0.98 ;  % Y
              -0.32  -0.36   0.12  -0.23  -0.05  0.22  0.21  -0.14 ]; % Z
end
```

9.2 Editing `ecgtransform.m`

The actual transformation between ECG to VCG is performed by the function `ecg_transform.m`. The new transformation matrix will have to be added to this file.

Start by opening `ecg_transform.m`:

```
function [X, Y, Z, VM] = ecgtransform(L1, L2, V1, V2, V3, V4, V5, V6, transform_matrix)

% Create matrix of leads V1, V2, V3, V4, V5, V6, I, II = Matrix E
E = [ V1, V2, V3, V4, V5, V6, L1, L2 ];

% Choose transformation matrix

switch transform_matrix
    case 'Dower'
        M = dowermatrix();
    case 'Kors'
        M = korsmatrix();
    otherwise
        error('Unknown transform_matrix: %s', transform_matrix);
end

...
```

Add a new `case` statement to the `switch` statement with a string that is the name of the transformation matrix (`'NewMatrix'`) and which assigns `M = newmatrix()` .

```
switch transform_matrix
    case 'Dower'
        M = dowermatrix();
    case 'Kors'
        M = korsmatrix();
    case 'NewMatrix'
        M = newmatrix();
    otherwise
        error('Unknown transform_matrix: %s', transform_matrix);
end
```

9.3 Editing `transform_mat.csv`

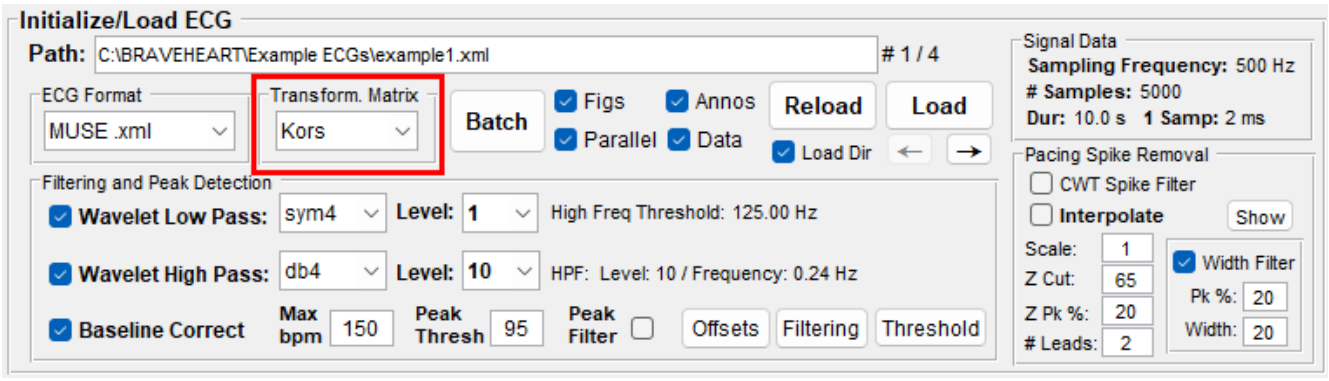
Finally, to add the new transformation matrix to the GUI dropdown, edit `transform_mat.csv` to include the new string used in the switch statement (in this example `NewMatrix`). Strings in `transform_mat.csv` need to be in the order they appear in the switch statement. As with editing the contents of all BRAVEHEART `.csv` files, do not include quotes around strings.

```
Kors
Dower
NewMatrix
```

9.4 Using Newly Added Transformation Matrices

If using the command line version of BRAVHEART (see **Chapter 13**), open `Annoparams.m` (or `Annoparams.csv` if you are using the compiled version) and assign the newly created transformation matrix string to the parameter `transformation_matrix_str` (e.g. `transformation_matrix_str = 'NewMatrix'`).

If you are using the GUI version of BRAVEHEART, simply choose the new format from the 'Transform. Matrix' dropdown (see GUI item 1.2 in **Chapter 12**).


Initialize/Load ECG
Path: C:\BRAVEHEART\Example ECGs\example1.xml
1 / 4
ECG Format
MUSE .xml
Transform. Matrix
Kors
Batch
Figs
Annos
Parallel
Data
Reload
Load
Load Dir
Filtering and Peak Detection
Wavelet Low Pass:
sym4
Level:
1
High Freq Threshold: 125.00 Hz
Wavelet High Pass:
db4
Level:
10
HPF: Level: 10 / Frequency: 0.24 Hz
Baseline Correct
Max bpm
150
Peak Thresh
95
Peak Filter
Offsets
Filtering
Threshold
Signal Data
Sampling Frequency: 500 Hz
Samples: 5000
Dur: 10.0 s 1 Samp: 2 ms
Pacing Spike Removal
CWT Spike Filter
Interpolate
Show
Scale:
1
Z Cut:
65
Z Pk %:
20
Leads:
2
Width Filter
Pk %:
20
Width:
20

10 Adding New ECG/VCG Parameters

New ECG/VCG measurements or parameters can easily be added to BRAVEHEART. To illustrate the process we will add a new measurement called `newvar` to BRAVEHEART. Note that although this parameter will be automatically added to BRAVHEART's output files, adding new measurements to the GUI output panels requires more extensive editing of the GUI figure and other functions that are beyond the scope of this guide.

10.1 Result Classes

BRAVEHEART currently has 4 results classes which are used to store different ECG/VCG measurements. These classes automatically update the output files at the time of data export, so minimal coding is required to add a new measurement. The 4 existing results class objects are somewhat arbitrarily divided into `VCG_Calc.m` which contains mostly ECG and VCG measurements such as ECG intervals, and vector measurements, `Lead_Morphology.m` which contains mostly ECG/VCG QRST complex measurements, `VCG_Morphology.m` which contains mostly VCG loop morphology measurements, and a minor results class `Beats_Stats` which contains calculations based on individual beats. We suggest that you add your new measurement to one of these existing results classes to minimize the need for coding. Each of these 4 results classes takes slightly different inputs, so make sure the class you add to has the required inputs for your new measurement. If your new function is extremely complicated and requires more than the ECG/VCG median beat signals, ECG median beat fiducial point annotations, or annotation parameters (which are the inputs to the existing results classes), addition of your parameter might require more extensive coding which is beyond the scope of this guide. Descriptions of the different calculations that are performed by each result class and the variable names which store the results internally and in output files are shown in **Chapter 24**. All results classes operate on median beats with the exception of `Beats_Stats` which is used to calculate variables based on the individual beats which make up the median beat.

We review the 4 results classes in the following sections:

10.1.1 `VCG_Calc.m`

`VCG_Calc.m` takes the following inputs:

- A `Beats` class object
- A `VCG` class object
- An `Annoparams` class object

A `Beats` class objects is used to describe individual beat annotations after first pass heuristic ECG annotation or median beat annotation. The `Beats` class object contains the following 7 properties which describe the locations of fiducial points and other features:

- `Q` – location (in samples) of QRS onset (Q_{on}) for the median beat
- `QRS` – location (in samples) of the R wave peak for the median beat
- `S` – location (in samples) of QRS offset (Q_{off}) for the median beat
- `T` – location (in samples) of T wave peak for the median beat
- `Tend` – location (in samples) of T wave offset (T_{off}) for the median beat
- `outlier` – index of outlier beats (not applicable for median beats)
- `pvc` – index of PVCs/non-dominant beats (not applicable for median beats)
- `QRS_rem_pvc` – location (in samples) of R wave peaks for beats that were removed after PVC detection (not applicable for median beats)
- `QRS_rem_outlier` – location (in samples) of R wave peaks for beats that were removed after outlier detection (not applicable for median beats)
- `QRS_rem_manual` – location (in samples) of R wave peaks for beats that were manually removed (not applicable for median beats)
- `QRS_rem_bad` – location (in samples) of R wave peaks for beats that were removed due to not being complete QRST complexes likely due to being too close to the start or end of the ECG signal or issues with overlapping beats (not applicable for median beats)

The data within a `Beats` object is accessed by querying its properties as if it were a normal data structure in MATLAB; for example, if you have an `Beats` object named `median_beat`, the location of the T wave end would be accessed as `median_beat.Tend`. This is critically important for any new functions that are added to the results classes, because the location of the fiducial points on the median beat signal need to be accounted for in all calculations; the median beat

signals have variable amounts of signal before and after the QRST complex which should NOT be included in calculations.

A `VCG` class object is analogous to the `ECG12` class object (see **Chapter 8.1**) and contains the following 7 properties:

- `hz` – the VCG sampling frequency in Hz
- `units` – an optional `string` describing VCG voltage units (usually `'mV'`)
- `X` – voltage data for VCG lead X
- `Y` – voltage data for VCG lead Y
- `Z` – voltage data for VCG lead Z
- `VM` – voltage data for the VM VCG lead ($\text{VM} = \sqrt{\text{X}^2 + \text{Y}^2 + \text{Z}^2}$).
- `endspikes` – data on location of pacemaker spikes (if present)

The data within a `VCG` object is accessed by querying its properties as if it were a normal data structure in MATLAB; for example, if you have an `VCG` object named `vcg`, lead X would be accessed as `vcg.X`.

An `Annoparams` class object is described in detail in **Chapter 6**. The data within an `Annoparams` object is accessed by querying its properties as if it were a normal data structure in MATLAB; for example, if you have an `Annoparams` object named `aps`, the parameter `lowpass` would be accessed as `aps.lowpass`.

In `VCG_Calc.m` the `VCG` class object is a variable named `v_uncropped`, the `Beats` class object is a variable named `medianbeat`, and the `Annoparams` class object is a variable named `aps`. `VCG_Calc` contains a function called `crop` which cuts out the median QRST complex from the full median VCG signal. The variable `v_cropped` is the `VCG` class object which only contains the actual QRST complex. `v_cropped.X` is equivalent to `v_uncropped.X(medianbeat.Q:medianbeat.Tend)` (and the same for leads Y, Z, and VM).

10.1.2 `Lead_Morphology.m`

`Lead_Morphology.m` includes the following inputs:

1. An `ECG12` class object
2. A `VCG` class object

3. A `Beats` class object
4. An `Annoparams` class object

The parameters in each of these class objects have been previously described above.

In `Lead_Morphology.m` the `ECG12` class object is a variable named `ecg`, the `VCG` class object is a variable named `vcg`, and the `Annoparams` class object is a variable named `aps`. When the `Lead_Morphology` class is called, the median beat `Beats` class object input is converted to a 1×4 vector named `fidpts`, which contains the values of `[beats.Q, beats.QRS, beats.S, beats.Tend]` in that order which is analogous to the values of [Q_{on}, R peak, Q_{off}, T_{off}]. The location of Q_{on} can therefore be called as `fidpts(1)`, the location of the R peak can be called as `fidpts(2)`, the location of Q_{off} can be called as `fidpts(3)`, and the location of T_{off} can be called as `fidpts(4)`. As another example, the QRS complex in lead X would be obtained as `vcg.X(fidpts(1):fidpts(3))`, and the full QRST complex in lead Y would be obtained as `vcg.Y(fidpts(1):fidpts(4))`. There is no `crop` function in `Lead_Morphology.m`, although it could be added if needed.

10.1.3 `VCG_Morphology.m`

`VCG_Morphology.m` includes the following inputs:

1. An `ECG12` class object
2. A `VCG` class object
3. A `Beats` class object

The parameters in each of these class objects have been previously described above.

In `VCG_Morphology.m` the `ECG12` class object is a variable named `ecg`, and the `VCG` class object is a variable named `vcg`. Similar to the `Lead_Morphology` class, when the `VCG_Morphology` class is called, the median beat `Beats` class object input is converted to a 1×4 vector named `fidpts` which contains the values of `[beats.Q, beats.QRS, beats.S, beats.Tend]` in that order (see **Chapter 10.1.2** above).

10.1.4 `Beats_Stats.m`

`Beats_Stats.m` includes the following inputs:

1. An `Beats` class object
2. time (in ms) per sample (`sample_time`)

In `Beats_Stats.m`, the `Beats` class object is a variable named `beats` and the milliseconds per sample (1/sampling frequency) is a numeric a variable named `sample_time`. Note that all of the other results classes operate on median beats, and the `Beats` class used as inputs to those results clases are annotations of the median beat. `Beats_Stats.m`, however, takes a `Beats` class object that includes all n beats used to create the median beat. For this reason, the `Beats_Stats.m` results class is a natural place to calculate variables that rely on beat-to-beat calculations such as heart rate variability metrics or if you want to measure something on all n beats individually. Unlike the other 3 results classes, processing of `Beats_Stats.m` cannot be disabled or bypassed if there is an error, so take care if you are adding some new calculation that could throw an error as this may completely prevent ECG processing. If you are going to be doing many more complex measurements on individual beats (which was not an original goal of BRAVEHEART) we would consider adding a new results class that can be disabled or bypassed if there are errors. This is not overly complicated, but the instructions for adding a new results class are beyond the scope of this user guide.

10.2 Adding a New Calculation

We will add our new measurement to `VCG_Calc.m`. We start by opening `VCG_Calc.m` and finding the `properties` section at the top.

```
classdef VCG_Calc
%
   properties (SetAccess=private)
     % SVG
     svg_x
     svg_y
     svg_z

     % SAI
     sai_x
     sai_y
     sai_z

     ...

     % QRS/QT intervals (in ms)
     qrs_int
     qt_int
   end
```

Further down in the `methods` section of `VCG_Calc.m` you will see a busy chunk of code starting around line 184 with the comment `% ADD NEW MEASUREMENTS IN THIS SECTION` assigning various values to the members of the `VCG_Calc` class:

```
% ADD NEW MEASUREMENTS IN THIS SECTION
...

   obj.vcg_length_qrs = curve_length(v_cropped.X, v_cropped.Y, v_cropped.Z, 1, qend);
   obj.vcg_length_t = curve_length(v_cropped.X, v_cropped.Y, v_cropped.Z, qend, length(v_cropped.X));
   obj.vcg_length_qrst = curve_length(v_cropped.X, v_cropped.Y, v_cropped.Z, 1, length(v_cropped.X));

...
```

In this code snippet (which is a small part of the `VCG_Calc` class property assignments), the values of `vcg_length_qrs`, `vcg_length_t`, and `vcg_length_qrst` (see **Chapter 24**) are assigned their respective properties in the class, `obj.vcg_length_qrs`, `obj.vcg_length_t`, and `obj.vcg_length_qrst`, respectively. Again, note how the calculations use the cropped VCG lead signals (just the QRST complex) (`v_cropped.X`, `v_cropped.Y`, and `v_cropped.Z`) rather than `v_uncropped` which is the QRST complex with additional signal before Q_{on} and additional signal after T_{off}.

Our new measurement `newvar` is calculated from a function called `calc_newvar(x,y,z,beats)` which is saved in a file `calc_newvar.m`:

```
newvar = calc_newvar(x,y,z,beats)
```

This function takes in median X, Y, and Z QRST complexes and a `Beats` object and outputs the value of `newvar`. We start by adding `newvar` to the `properties` section of `VCG_Calc.m`:

```
classdef VCG_Calc
%
   properties (SetAccess=private)
     % SVG
     svg_x
     svg_y
     svg_z

     % SAI
     sai_x
     sai_y
     sai_z

     % New Measurement
     newvar
     ...

     % QRS/QT intervals (in ms)
     qrs_int
     qt_int
   end
```

We then insert the function which is used to calculate `newvar` into the `methods` section of `VCG_Calc.m`, assigning its output to `obj.newvar`:

```
% ADD NEW MEASUREMENTS IN THIS SECTION
...

   obj.vcg_length_qrs = curve_length(v_cropped.X, v_cropped.Y, v_cropped.Z, 1, qend);
   obj.vcg_length_t = curve_length(v_cropped.X, v_cropped.Y, v_cropped.Z, qend, length(v_cropped.X));
   obj.vcg_length_qrst = curve_length(v_cropped.X, v_cropped.Y, v_cropped.Z, 1, length(v_cropped.X));

   obj.newvar = calc_newvar(v_cropped.X, v_cropped.Y, v_cropped.Z, beats);

...
```

Once the new variable is added to a result class, no further files require editing. The functions for exporting data will now automatically include a variable called `newvar` and assign it the value calculated by the `calc_newvar` function.

Chapter 24 lists the different parameters that are included in each of the results classes.

11 Adding New Denoising/Filtering

BRAVEHEART utilizes wavelet based high-frequency and low-frequency denoising/filtering (see **Chapter 7** which have many benefits over conventional linear filters [1, 8]. However, if you have need for an alternative denosing/filtering scheme, this can be added to BRAVEHEART by editing the `ecgfilter.m` and `ECG12.m` files.

11.1 Calling the ECG Denoising/Filtering Function from `ECG12.m`

Lines 153–180 of `ECG12.m` (see **Chapter 8** for further details on the `ECG12` class) are devoted to the denoising/filtering class function called `filter`. There is no analogous denoisng/filtering function within the `VCG` class because the ECG is filtered prior to transformation into a VCG.

```
function [obj, highpass_lvl_min] = filter(obj, maxRR_hr, aps)

...
[obj.I, obj.II, obj.III, obj.avR, obj.avF, obj.avL, obj.V1, obj.V2, obj.V3, obj.V4, obj.V5, obj.V6,
    highpass_lvl_min] = ...
        ecgfilter(obj.I, obj.II, obj.III, obj.avR, obj.avF, obj.avL, obj.V1, obj.V2, obj.V3, obj.V4,
            obj.V5, obj.V6, ...
        obj.hz, maxRR_hr, ...
        aps.lowpass, aps.wavelet_level_lowpass, aps.wavelet_name_lowpass, ...
        aps.highpass, aps.wavelet_level_highpass, aps.wavelet_name_highpass);
...
end
```

This calls the function `ecgfilter()` on the `ECG12` object. If you have an `ECG12` class object named `ecg`, the filtered ECG class object is generated as:

```
filtered_ecg = ecg.filter(NaN, aps);
```

where `aps` is an `Annoparams` object (see **Chapter 6**) that includes the filtering settings which are passed into the function. The first `NaN` is a placeholder for some additional high-pass filtering settings that may be incorporated in a future release of BRAVEHEART, but for now the value has no effect. When using the GUI there is a second optional output (`highpass_lvl_min`) that is used to show the level used for high-pass wavelet decomposition in the GUI (see line 3233 in `braveheart_gui.m`).

11.2 Editing `ecgfilter.m`

To add new filtering schemes you will need to edit the function `ecgfilter()` in the file `ecgfilter.m`:

```
function [L1, L2, L3, avR, avF, avL, V1, V2, V3, V4, V5, V6, final_lf_wavelet_lvl_min] = ...
    ecgfilter(L1, L2, L3, avR, avF, avL, V1, V2, V3, V4, V5, V6, freq, maxRR_hr, ...
    wavelet_filt, wavelet_level, wavelet_name, wavelet_filt_lf, wavelet_level_lf, wavelet_name_lf)
```

This function takes the inputs of the 12 ECG leads individually (`L1`, `L2`, `L3`, `avF`, `avL`, `avR`, `V1`, `V2`, `V3`, `V4`, `V5`, `V6`), the sampling frequency `freq`, the placeholder variable for future functionality as noted above (`maxRR_hr = NaN`), and variables which describe if high/low pass filtering should be performed, and if so, which mother wavelet and decomposition level should be used (`wavelet_filt`, `wavelet_level`, `wavelet_name`, `wavelet_filt_lf`, `wavelet_level_lf`, `wavelet_name_lf`).

The outputs of the function include the 12 filtered leads, and a numeric variable `final_lf_wavelet_lvl_min` which encodes the level of decomposition used for high-pass denoising/filtering that is used for GUI display only.

Our suggestion would be to **NOT** alter the output variable structure of the `ecgfilter` function so that there is still an numeric output variable called `final_lf_wavelet_lvl_min` even if it is not being used. This will save you some trouble integrating with the GUI even if you are utilizing a new filtering method that has nothing to do with wavelets, although this is not an issue if you are using the command line version of BRAVEHEART. Updating the GUI to reflect use of the new filtering scheme is beyond the scope of this guide.

Edit `ecgfilter.m` as needed to implement your new filtering scheme with the requisite outputs as noted above. To control your new filter you can edit `Annoparams.m` and simply add any new parameters and default values that you want. These parameters will obviously not be editable in the GUI unless you edit the GUI figure/code. You can then edit the inputs as needed when calling the filtering function within `ECG12.m` at the location noted above. The only place `ecgfilter` is called within the entire BRAVEHEART software package is within `ECG12.m`, so you will not have to edit any other sections of code to fully integrate the new filtering function with BRAVEHEART. You should leave the parameters reflecting the nominal wavelet based denoising/filtering in `Annoparams.m` because the output files will throw errors if these values are undefined when using the GUI. Any new parameters that are added to `Annoparams.m` will automatically be exported with their values during ECG processing.

For example, lets add a simple low-pass filter with a passband frequency of 150 Hz and a simple high-pass filter with a passband frequency of 0.2 Hz. **NOTE:** these simple types of filters will likely cause significant phase shifts in the signal and are not recommended for actual use! - this is just an illustrative example of how to edit the BRAVEHEART denoising/filtering source code.

We first add new parameters to `Annoparams.m` to control our new filters:

```
% New filtering parameters
newlowpass = 1;          % Low-pass filter on/off
newhighpass = 1;         % High-pass filter on/off
lowpassfreq = 150;        % Low-pass passband (Hz)
highpassfreq = 0.2;       % High-pass passband (Hz)
```

Next, we edit `ecgfilter.m` so that it looks like this (remember we left `final_lf_wavelet_lvl_min` in the output to avoid breaking the GUI). In the interest of space all 12 leads were not included in this code snippet:

```
function [L1, L2, L3, avR, avF, avL, V1, V2, V3, V4, V5, V6, final_lf_wavelet_lvl_min] = ...
    ecgfilter(L1, L2, L3, avR, avF, avL, V1, V2, V3, V4, V5, V6, freq, ...
    newlowpass, lowpassfreq, newhighpass, newhighpassfreq)

% High-pass
if newhighpass == 1
        L1 = highpass(L1, highpassfreq, freq);
    ...
        V6 = highpass(V6, highpassfreq, freq);
end

% Low-pass filter
if newlowpass == 1
        L1 = lowpass(L1, lowpassfreq, freq);
    ...
        V6 = lowpass(V6, lowpassfreq, freq);
end
        final_lf_wavelet_lvl_min = 0;       % This needs a value to not break GUI as noted above

end
```

11.3 Editing `ECG12.m`

Finally, we edit `ECG12.m` starting at line 164 to incorporate the new `ecgfilter` function into the `ECG12` class function `filter`, leaving `highpass_lvl_min` in the output of the function to avoid breaking the GUI as noted above. Note the need to use the prefix `obj.` to assign these values to the `ECG12` class object.

```
function [obj, highpass_lvl_min] = filter(obj, maxRR_hr, aps)

[obj.I, obj.II, obj.III, obj.avR, obj.avF, obj.avL, obj.V1, obj.V2, obj.V3, obj.V4, obj.V5, obj.V6,
   highpass_lvl_min] = ...
        ecgfilter(obj.I, obj.II, obj.III, obj.avR, obj.avF, obj.avL, obj.V1, obj.V2, obj.V3, obj.V4,
           obj.V5, obj.V6, ...
        obj.hz, ...
        aps.newlowpass, aps.lowpassfreq, aps.newhighpass, aps.highpassfreq);
...
end
```

Enabling/disabling these new high- and low-pass filters, and adjusting their frequency passbands, is now controlled via setting the parameters in `Annoparams.m`. Although the default wavelet-based denoising/filtering parameters are still present in `Annoparams.m` these parameters will be ignored and the new filtering scheme will be used.

12 Using the BRAVEHEART GUI

12.1 GUI Overview

In addition to batch processing of ECGs via `braveheart_batch.m` (see **Chapter 13**), the BRAVEHEART GUI facilitates control over each step of ECG/VCG processing and can generate additional figures and data visualizations. In this chapter we provide an overview of the different functions of the GUI and how to process ECGs from files to measurements saved for further processing.

To run the BRAEHEART GUI, type `braveheart_gui` in the MATLAB Command Window if the program is being run via source code within MATLAB, or open the executable if running via a compiled version of the GUI.

The BRAVEHEART GUI window immediately after opening is shown below in **Figure 1**. The appearance on your computer may vary slightly based on your operating system. A screen resolution of at least 1920 x 1080 is required.

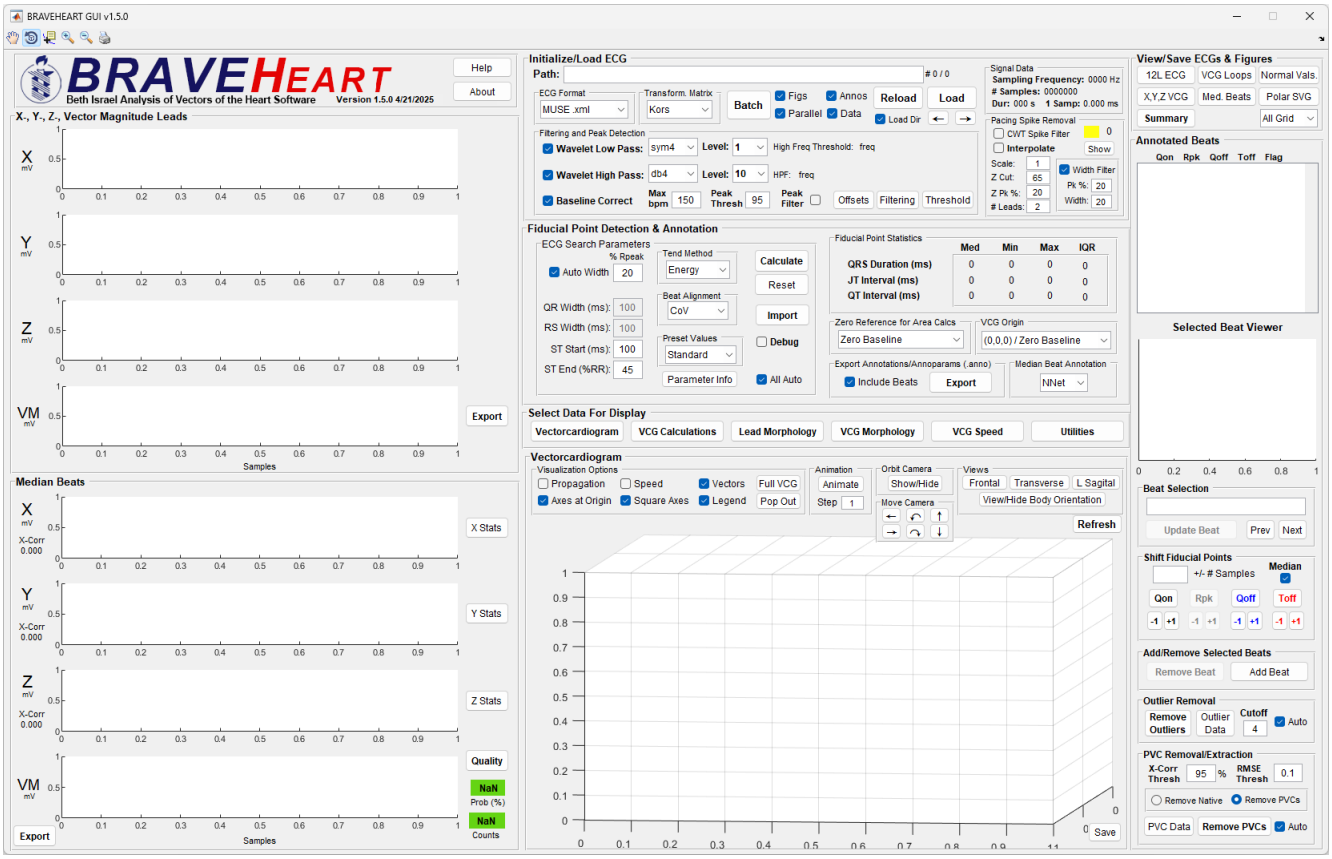


Figure 1: GUI when BRAVEHEART is first loaded.

The GUI is organized into 10 different sections which, in general, have different functions. The

different sections of the GUI are shown below in **Figure 2**; a general description of each section is provided followed by detailed descriptions of GUI items within each section.

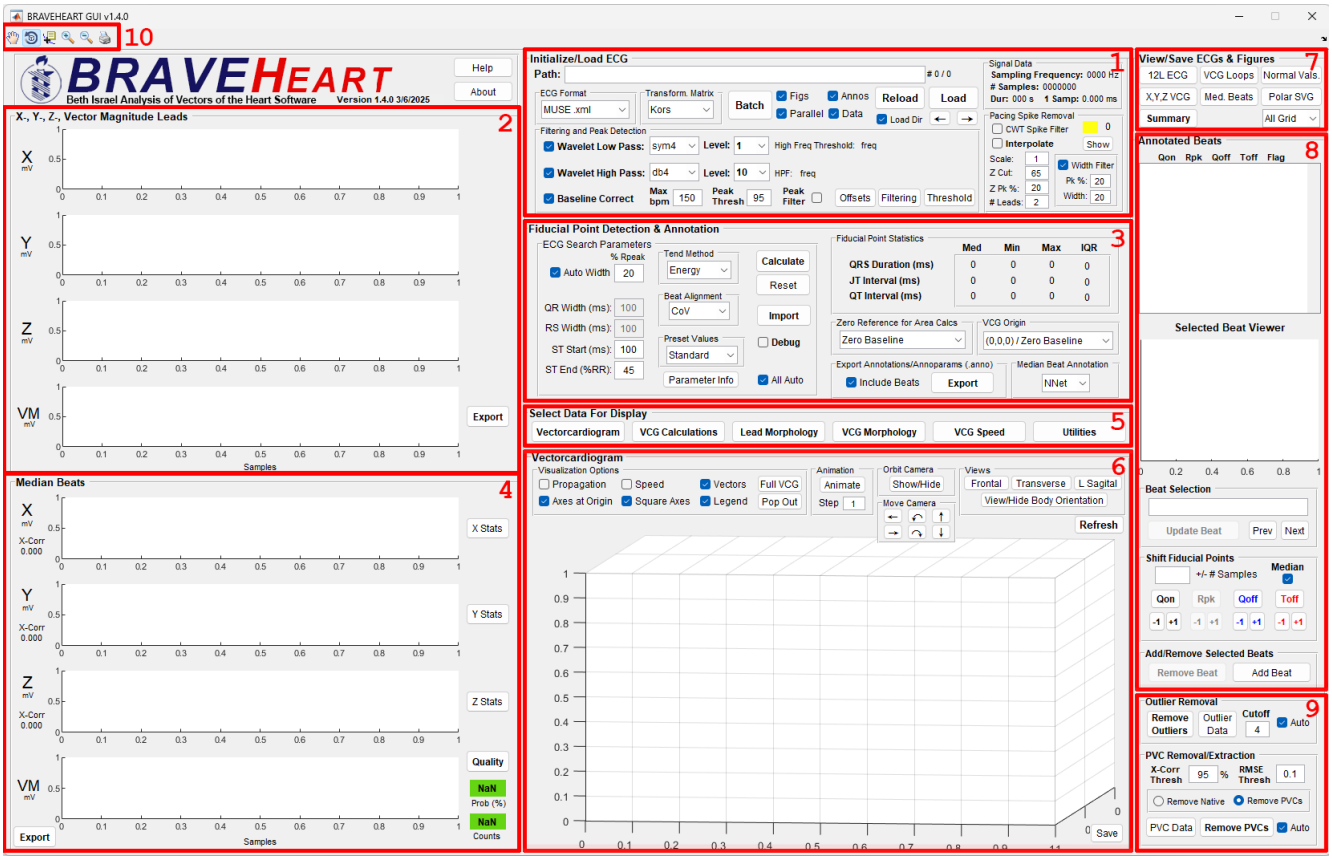


Figure 2: GUI Sections 1–10.

Many of the parameters that are set using the GUI correspond to the properties in the `Annoparams.m` class, and users are referred to **Chapter 6** for further details of each property. A summary of the link between `Annoparams.m` and sections of the GUI is provided in **Chapter 29**. The GUI also contains other functions and visualization that are unique to the GUI and which allow fine control and/or visualization of different steps in ECG/VCG processing.

Throughout the guide, to denote a specific section/item within the GUI, the term 'GUI item N.M' with numbers in red text will be used; N refers to the GUI section (1–10) as noted in **Figure 2** above, and M refers to the numbered item within the section as detailed in the subsequent sections. For example, the **Load** button would be denoted as GUI item **1.9** as it is item **9** in GUI section **1**, and the **Calculate** button would be similarly denoted as GUI item **3.6**.

Section 6 is unique in that what is displayed depends on what button is selected in Section 5. For Section 6 alone, there is an extra number to denote the specific part of the GUI that is being shown. For example, the **Vectorcardiogram** button (GUI item **5.1**) loads GUI section 6.1, and the first item in this section is noted as GUI item **6.1.1**, while the the **VCG Morphology** button

loads GUI section 6.4 and the first item in this section is noted as GUI item 6.4.1.

12.2 GUI Section 1 – ECG Loading, Filtering, R Peak Detection, and Baseline Correction

Section 1:

Choose an ECG file and load (or reload) the ECG data. Choose options for transformation of the ECG into a VCG, QRS peak detection, signal filtering, and pacemaker spike removal.

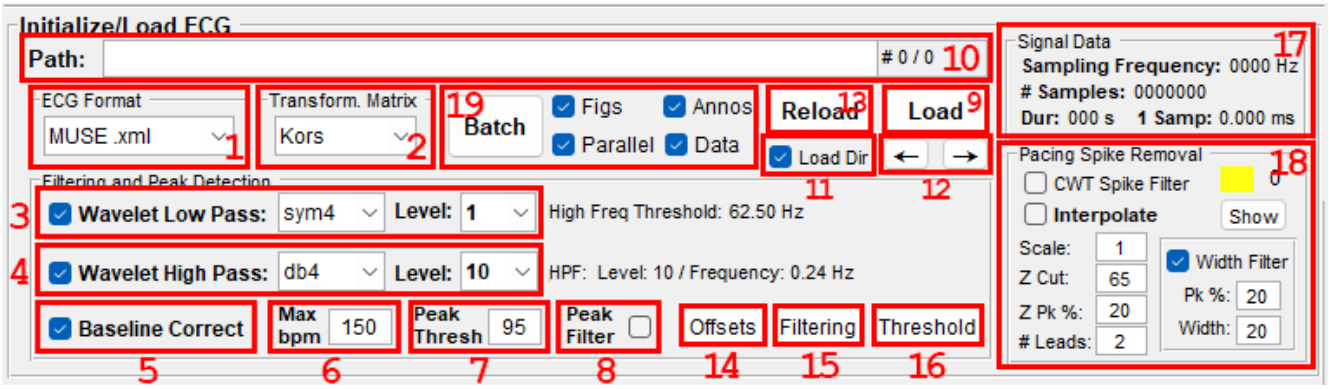


Figure 3: GUI Section 1

1. Dropdown to select the format of the ECG to be loaded. Sets the value of `format` in the ECG constructor `ECG12.m`. The link between the text shown in the dropdown and the actual `format` that is utilized by the GUI is contained in the external file `ecg_formats.csv` (see **Chapter 8**). Additional information on ECG formats that can be read by BRAVEHEART and how to add new formats can be found in **Chapter 23** and **Chapter 8**, respectively.

---

2. Dropdown to select the VCG transformation matrix used to transform the ECG into a VCG. Sets the value of `transform_matrix_str` in `Annoparams.m`. BRAVEHEART includes the Kors [4] and inverse Dower [5] transformation matrices. Additional transformation matrices can be added as described in **Chapter 9**.

---

3. Select and choose options for low-pass filtering. The ☑ Wavelet Low Pass checkbox enables/disables low-pass filtering by setting `lowpass = 1` (checked) or `lowpass = 0` (unchecked) in `Annoparams.m`. The dropdowns allows you to choose the wavelet used for low-pass filtering (`wavelet_name_lowpass`) and the level of filtering (`wavelet_level_lowpass`) in `Annoparams.m` (see **Chapter 6.2** and **Chapter 7**).

After the ECG is loaded, the text to the right of GUI item **1.3** shows the low-pass cutoff frequency which corresponds to the chosen wavelet level based on the equation $f_s/(2^{n+1})$ where f_s is the ECG sampling frequency and n is the chosen level (`wavelet_level_lowpass`). In this example, the low-pass filter has a cutoff frequency of 62.5 Hz which corresponds to level 2 at a sampling frequency of 500 Hz. In general try to avoid choosing a level that results in low-pass filtering cutoff < 40 Hz as the QRST complex tends to include frequencies between 0.5-40 Hz [3]. Filtering can be visualized with a figure generated with the Filtering button (GUI item **1.15**) as shown in **Figure 65**.

---

4. Select and choose options for high-pass filtering. The ☑ Wavelet High Pass checkbox enables/disables high-pass filtering by setting `highpass = 1` (checked) or `highpass = 0` (unchecked) in `Annoparams.m`. The dropdowns allow you to choose the wavelet used for high-pass filtering (`wavelet_name_highpass`) and the level of filtering (`wavelet_level_highpass`) in `Annoparams.m` (see **Chapter 6.2** and **Chapter 7**).

   After the ECG is loaded, the text to the right of GUI item **1.4** shows the high-pass cutoff frequency which corresponds to the chosen wavelet level based on the equation $f_s/(2^{n+1})$ where f_s is the ECG sampling frequency and n is the chosen level (`wavelet_level_highpass`). In this example, the high-pass filter has a cutoff frequency of 0.24 Hz which corresponds to level 10 at a sampling frequency of 500 Hz. In general try to avoid choosing a level that results in high-pass filtering cutoff > 0.5 Hz as the QRST complex tends to include frequencies between 0.5-40 Hz [3]. Filtering can be visualized with a figure generated with the Filtering button (GUI item **1.15**) as shown in **Figure 65**.

---

5. Checkbox to enable/disable physiological baseline correction. This part of ECG/VCG processing adjusts the baseline after filtering to remove any residual offsets so the TP segment (or the TQ segment in atrial fibrillation) of the ECG signal approximates 0 mV as this segment of the ECG is truly isoelectric. The ☑ Baseline Correction checkbox sets `baseline_correct_flag = 1` (checked) or `baseline_correct_flag = 0` (unchecked) in `Annoparams.m`. Setting the 0 mV reference correctly is *critical* for accurately calculating areas under median QRST complexes. Further details about baseline correction can be found in **Chapter 15.3**, **Figure 38**, **Figure 39**, and **Figure 40**.

---

6. Textbox to set `maxBPM` in `Annoparams.m` for QRS peak detection. See **Chapter 15.1** and **Figure 24** for additional details of how this parameter controls QRS peak detection.

---

7. Textbox to set `pkthresh` in `Annoparams.m` for QRS peak detection. See **Chapter 15.1** and **Figure 24** for additional details of how this parameter controls QRS peak detection.

---

8. Checkbox to enable/disable VM filtering prior to R peak detection. The ☑ Peak Thresh checkbox sets `pkfilter = 1` (checked) or `pkfilter = 0` (unchecked) in `Annoparams.m`. In general, this option will not be needed for most ECGs. If peak filtering is enabled, you have have to adjust the value of `pkthresh` (GUI item 1.7) to avoid missing peaks or detecting low amplitude noise. See **Chapter 15.1**, **Figure 24**, and **Figure 25** for additional details of how this parameter controls QRS peak detection.

---

9. The **Load** button opens a dialog box so the user can choose an ECG file to process. If the ☑ Load Dir checkbox (GUI item 1.11) is checked, the load process also loads a list of all files in the same directory with the same extension as the loaded file so the user can quickly cycle through multiple files in a directory using GUI item 1.12 (see below).

---

10. Textbox which shows the full path of the ECG file which has been loaded. This can be manually edited; to load the file after manually editing the file path click the **Reload** button (See GUI item 1.13) below. To the right of the textbox there are 2 numbers separated by a /. If the ☑ Load Dir checkbox (GUI item 1.11) is checked, the load process also loads a list of all files in the same directory with the same extension as the loaded file, and the numbers reflect the number of the current ECG out of the total number of ECGs which can be cycled through using GUI item 1.12 (see below). If the ☐ Load Dir checkbox (GUI item 1.11) is unchecked, only a single ECG is loaded, the user cannot cycle through ECGs using GUI item 1.12, and the text defaults to "1 / 1".

---

11. The ☑ Load Dir checkbox loads an alphabetized list of all files of the same format within the directory the chosen file is loaded from to allow use of the ← and → buttons (GUI item 1.12) to easily cycle through files within a directory. If ☐ Load Dir if unchecked, only a single file is loaded and the ← and → buttons have no effect. Checking ☑ Load Dir without subsequently loading an ECG using the **Load** button (GUI item 1.9) has no effect as the list of files is loaded when the callback triggered by **Load** is activated. If there are a very large number of ECG files in the directory, disabling ☐ Load Dir may significantly speed up loading an individual ECG.

---

12. The ← and → buttons cycle through all of the files with the same extension as the file chosen with the **Load** button if ☑ Load Dir is also checked as noted in GUI item 1.11

above. Files are sorted alphabetically. As you cycle through the files, the number of the file will change (see GUI item **1.11** above).

---

13. After the ECG has been initially loaded using the **Load** button, the **Reload** button will reload the current ECG from scratch without having to select the file again. The **Reload** button is also used after manually editing the file path textbox (GUI item **1.10** above) to load the file in the path textbox. The **Reload** button does NOT reset the GUI parameters to the default values in `Annoparams.m`; this function is performed by the Reset button (GUI item **3.7**).

---

14. The Offsets button displays a figure that shows the baseline offsets in all leads and how filtering and baseline correction have performed physiological re-zeroing of the ECG signal voltage. See **Figure 40** for additional information.

---

15. The Filtering button displays a figure that shows the effect of the selected filtering on the ECG signal. See **Figure 65** for additional information.

---

16. The Threshold button displays a figure that shows how the parameters `pkthresh` and `maxBPM` are working to identify R peaks in the VM lead as part of first pass VCG fiducial point processing. Further details are available in **Chapter 15.1** and **Figure 24**.

---

17. Summary of ECG signal data that was loaded, including the sampling frequency, number of ms between samples, and signal duration in seconds and samples.

---

18. Adjust parameters for automatic pacemaker spike/artifact removal. The inset box with the ☑ Width Filter checkbox activates/deactivates the median filter method of pacemaker spike detection by setting `spike_removal = 1` (checked) or `spike_removal = 0` (unchecked) in `Annoparams.m`. The 'Peak %' textbox sets the value of `pacer_thresh` and the 'Pk Width' textbox sets the value of `pacer_spike_width` in ms in `Annoparams.m`. Further details are can found in **Chapter 15.2**.

The remainder of the section is used to control the CWT method of pacemaker spike removal. The ☑ CWT Spike Filter checkbox enables/disables the CWT method of pacemaker spike removal by setting `cwt_spike_removal = 1` (checked) or `cwt_spike_removal = 0` (unchecked) in `Annoparams.m`. The ☑ **Interpolate** checkbox enables/disables pacemaker spike interpolation by setting `interpolate = 1` (checked) or `interpolate = 0` (unchecked) in `Annoparams.m`. The 'Scale' textbox sets the value of

`pacer_maxscale`, the 'Z Cut' textbox sets the value of `pacer_zcut`, the 'Z Pk %' textbox sets the value of `pacer_zpk`, and the '# Leads' textbox sets the value of `pacer_spike_num`, all in `Annoparams.m`. Further details are can found in **Chapter 15.2**.

The Show button generates figures showing results of pacemaker spike detection and interpolation if the CWT method is being used. Further details can be found in **Chapter 15.2** and **Chapter 20**.

The number next to the yellow square reflects the number of leads in which pacing was detected. If this number is greater than `pacer_spike_num`, the yellow box will appear with a black electrical bolt. If pacing is detected in at least 1 lead but the number of leads with pacing is less than `pacer_spike_num`, the black electrical bolt appears with a white background instead. If no pacing is detected the square is not displayed:

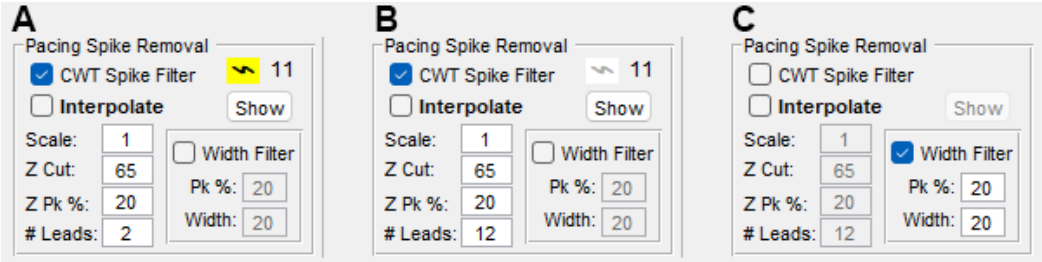


Figure 4: Changes in the GUI depending on how many leads had pacing detected. **A:** Pacing is detected in $\geq$ than `pacer_spike_num` leads. The number next to the electric bolt on a yellow background in the number of leads (out of 12) in which pacing was detected. **B:** Pacing is detected in $<$ `pacer_spike_num` leads. The number next to the electric bolt on a white background in the number of leads (out of 12) in which pacing was detected. In this case, `pacer_spike_num` was set to 12 but pacing spikes were only detected in 11 leads. **C:** CWT method is not being used or pacing is not detected. There is no number of leads displayed and no electric bolt displayed.

---

19. Settings for running `BRAVEHEART_batch()`. The **Batch** button brings up a dialog box that allows the user to choose a folder of ECGs for batch processing. Checkboxes control output of batch processing. Further details of batch processing via the GUI can be found in **Chapter 14**.

12.3 GUI Section 2 – VCG Display

Section 2:

Display the X, Y, Z, and VM leads for the loaded 12-lead ECG after processing based on parameters set in GUI Section 1.

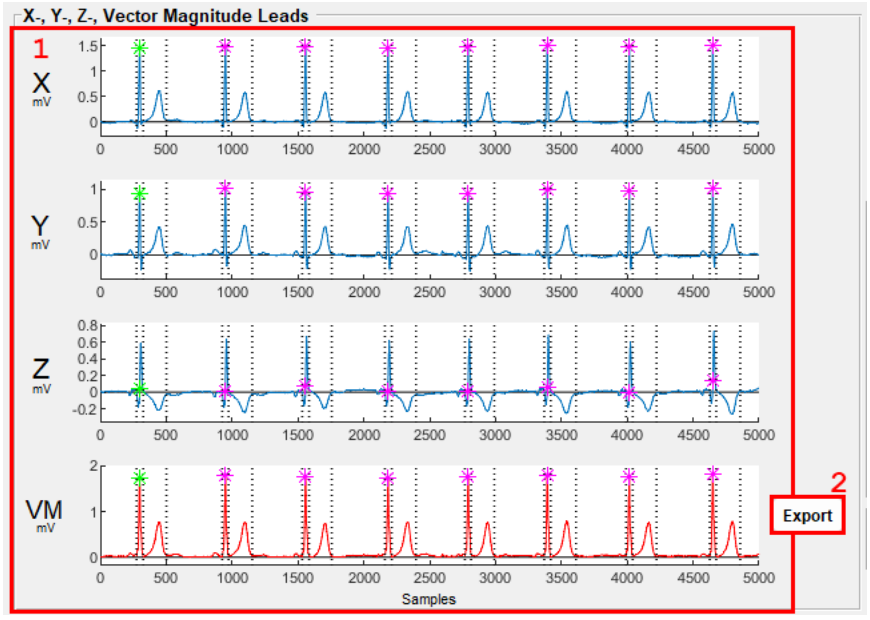


Figure 5: GUI Section 2

1. Display of the filtered and baseline corrected VCG (X, Y, and Z leads) and VM leads. Detected QRS peaks are shown with a magenta * with the exact location based on the VM lead R wave peak. After first pass annotation is complete (user clicked **Calculate** – GUI item **3.6**), dashed black lines show the locations of Q_{on}, Q_{off}, and T_{off} (see **Figure 6** below). When a beat is selected in the '**Selected Beat Viewer**' (GUI item **8.2**), the magenta * is changed to a green * as shown in **Figure 6** below where the first beat has been selected:

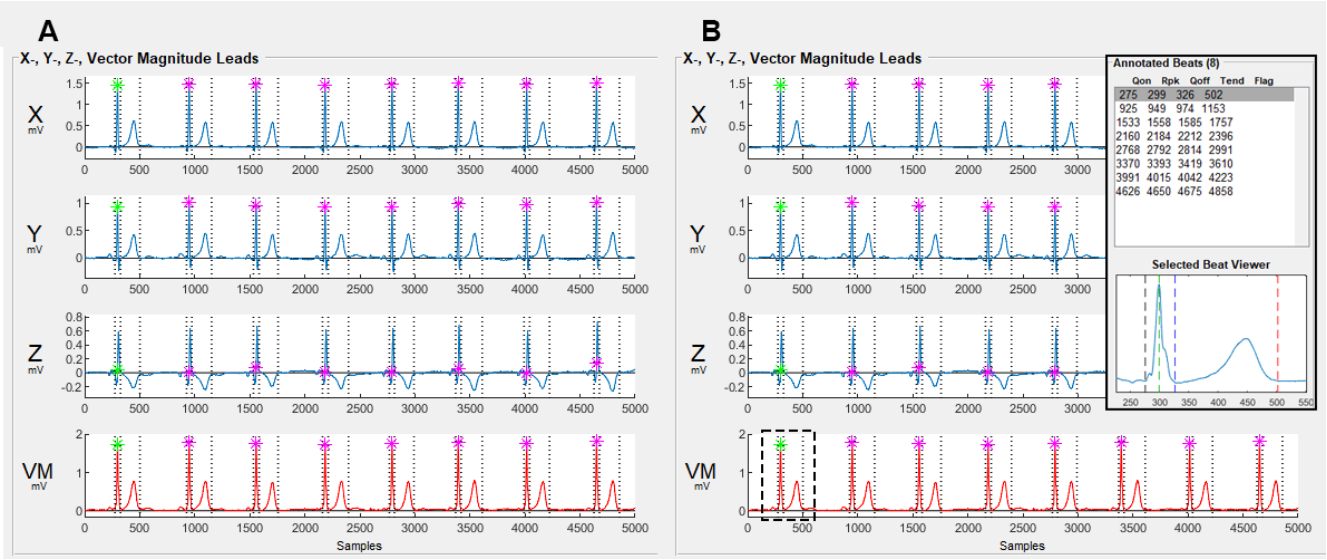


Figure 6: **A:** The * corresponds to each detected R peak in the VM lead. After first pass annotation, dashed vertical lines mark the Q_{on}, Q_{off}, and T_{off} for each beat. **B:** When a beat is selected in the '**Annotated Beats**' section (GUI item 8.1), the selected beat (dashed rectangle) is highlighted with a green * and displayed in the '**Selected Beat Viewer**' in GUI item 8.2 (inset panel).

Beats that are identified as 'PVCs' or other non-dominant QRST morphologies (see **Chapter 15.4**) will be marked with a triangle around the * (⚠), and beats that are identified as 'outliers' will be marked with a circle around the * (⊛). These symbols correspond to the '#' and '**' symbols in the 'Annotated Beats' section of Section 8 (GUI item 8.1). Note that a beat can be labeled as both a "PVC" and an "outler", in which case it is labeled with both a triangle *and* circle in the '**X-, Y-, Z-, Vector Magnitude Leads**' section, and with both a '#' and '**' in the '**Annotate Beats**' list. See **Chapter 15.4** and **Chapter 15.5** for further information.

---

2. the Export button exports the VCG signal data (X, Y, Z, and VM leads) as `.mat` and `.csv` files. The exported files will appear in the same directory as the ECG file with the naming convention `<filename>_xyz_beats.mat` and `<filename>_xyz_beats.csv`. The `.mat` file contains a structure `xyz_beats` that contains a $n \times 4$ matrix with columns corresponding to the X, Y, Z, and VM leads in that order, and each row corresponding to a sample. The `.csv` file contains 4 columns corresponding to the X, Y, Z, and VM leads in that order. In both file formats, exported signals are in units of mV.

12.4 GUI Section 3 – Fiducial Point Annotation and Median Beat Creation/Annotation

Section 3:

Choose options for first pass fiducial point annotation and median beat creation and annotation.

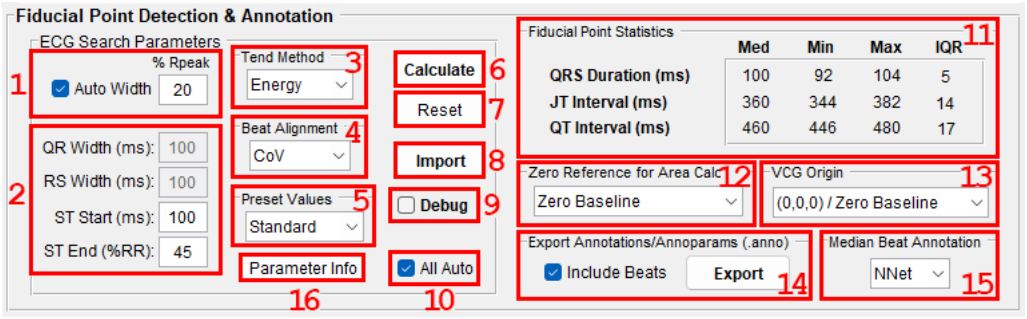


Figure 7: GUI Section 3

1. Select and set options for using automatic QRS width estimation during first pass heuristic annotation. ☑ Auto Width sets the value of `autoMF = 1` (checked) or `autoMF = 0` (unchecked) in `Annoparams.m`. The textbox '% Rpeak' sets the value of `autoMF_thresh` in `Annoparams.m`. If `autoMF = 1` the first pass annotation algorithm dynamically estimates the QRS search window for each beat individually rather than using a fixed window as specified by `QRwidth` and `RSwidth` (see GUI item 3.2 below). Further details of annotation parameters can be found in **Chapter 6.2**, **Chapter 18**, and **Figure 45**.

---

2. Textboxes to set the value of `QRwidth`, `RSwidth`, `STstart`, and `STend` in `Annoparams.m`. Note that `QRwidth` and `RSwidth` can only be changed if ☐ Auto Width (GUI item 3.1) is unchecked, as otherwise the `QRwidth` and `RSwidth` are dynamically set for each beat individually based on QRS width estimation, which is usually the best option. For further details on these parameters and their relation to fiducial point annotation, see **Chapter 6.2**, **Chapter 18**, and **Figure 45**.

---

3. Dropdown to select the method of first pass T wave end (T_{off}) annotation by setting `Tendstr` in `Annoparams.m`. The nominal value of 'Energy' should be used in most cases as this method is more robust to abnormal T wave morphologies than are the other options of 'Tangent' (which uses the tangent method) and 'Baseline' (which finds the baseline crossing). Further details can be found in **Chapter 6.2**, **Chapter 18**, and **Figure 45**.

---

4. Dropdown to select the method of beat alignment for median beat construction by setting `align_flag` in `Annoparams.m`. 'CoV' uses the filtered center of voltage of the QRS

complex to align beats, while 'Rpeak' uses the location of the R wave peak voltage to align beats. See **Chapter 6.2** for additional details.

---

5. Dropdown to select preset values for `pkthresh`, `maxBPM`, `QRwidth`, `RSwidth`, `STstart`, and `STend`. This can be useful when you have certain types of ECGs (such as ECGs with very long QT intervals or frequent PVCs) and you want to quickly adjust the annotation settings and save them for future use. The dropdown pulls values from the external file `search_presets.csv`. Further details can be found in **Chapter 21**.

---

6. The **Calculate** button begins processing of the ECG/VCG after it has been loaded and filtered. Press this once all filtering and annotation parameters are set appropriately.

---

7. The Reset button resets all GUI parameters to their nominal values which are pulled from `Annoparams.m`.

---

8. The **Import** button overrides the set GUI/annotation parameters and pulls annotation parameters and fiducial point annotations from an external `.anno` file. Further details can be found in quick start Example 5 (**Chapter 5.5**) and **Chapter 17**.

---

9. Checkbox to enable/disable debug figures for both first pass and median beat annotation. The ☑ Debug checkbox sets `debug = 1` (checked) or `debug = 0` (unchecked) in `Annoparams.m`. Further details on using debugging can be found in **Chapter 31.8**, **Figure 45**, and **Figure 70**.

---

10. Checkbox to easily enable/disable automatic outlier and automatic PVC removal using a single click. When ☑ All Auto is checked, the ☑ Auto checkboxes for both outlier removal (GUI item 9.4), and PVC removal (GUI item 9.10) are also checked, and `pvc_removal = 1` and `outlier_removal = 1` in `Annoparams.m`, thus enabling automatic outlier and PVC removal during ECG/VCG processing. When ☐ All Auto is unchecked, the ☐ Auto checkboxes for outlier removal and PVC removal both get unchecked, disabling both automatic outlier removal and automatic PVC removal (`pvc_removal = 0` and `outlier_removal = 0` in `Annoparams.m`).

---

11. Summary of first pass annotation statistics. The median ('Med'), minimum ('Min'), maximum ('Max'), and interquartile range ('IQR') are displayed. This is one way of quickly assessing the quality and consistency of first pass annotation.

---

12. Dropdown to set `baseline_flag` in `Annoparams.m`. In general, this should not be changed from the nominal value of 'Zero Baseline' if ECGs are filtered and baseline corrected appropriately. Further details can be found in **Chapter 6.2**.

---

13. Dropdown to set `origin_flag` in `Annoparams.m`. In general, this should not be changed from the nominal value of 'Zero Baseline' if ECGs are filtered and baseline corrected appropriately. Further details can be found in Further details can be found in **Chapter 6.2**.

---

14. This section is used to export annotation (extension `.anno`) files. The **Export** button generates an `.anno` file that contains the annotation parameters used to process the file. If the ☑ Include Beats checkbox is checked, the `.anno` file also contains information on the beats used to construct the median beat. See quick start Example 5 (**Chapter 5.5**) and **Chapter 17** for additional information.

---

15. Sets the method of annotating the median beat using either the standard heuristic annotator 'Std' or the custom BRAVEHEART neural network (NN) median beat annotater 'NNet'. In almost all cases the nominal value of 'NNet' should be used as it is the most accurate way to annotate median beats. In rare cases where the NN fails to accurately annotate a median beat, switching to 'Std' may work, although usually ECGs that fail NN annotation are ECGs with significant noise and artifact that are difficult to annotate regardless of method. Details of neural network development and annotation performance are available in the methods manuscript [1] and **Chapter 19**.

---

16. The Parameter Info button opens a file which contains information on annotation parameters and heuristic first pass annotation.

12.5 GUI Section 4 – Median Beat Display and Quality Assessment

Section 4:

Visualize median X, Y, Z, and VM beats, the individual beats that make up the median beat, and information about the quality of ECG processing, median beat construction, and median beat annotation.

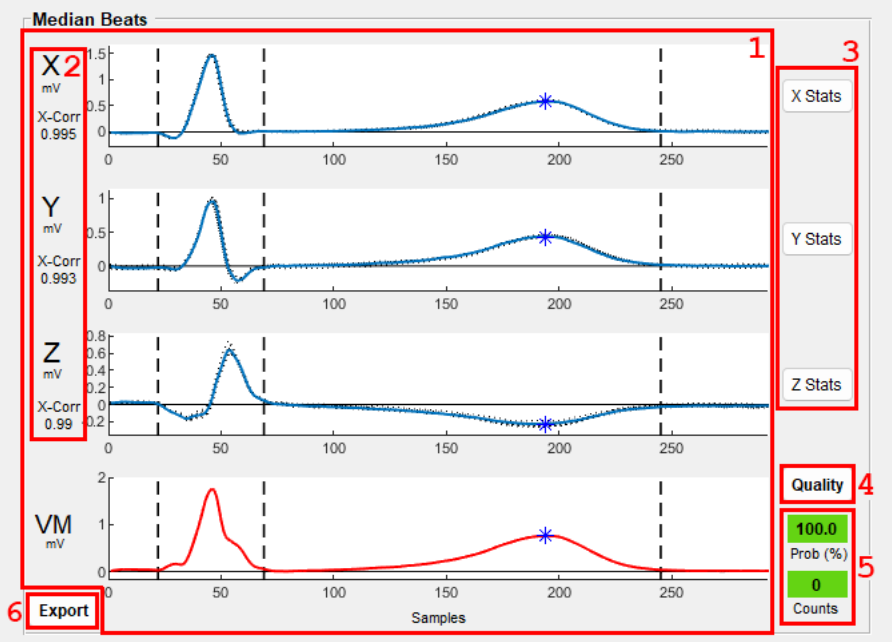


Figure 8: GUI Section 4

1. Display of the X, Y, Z, and VM median beats. The actual median beats are shown in blue for leads X, Y, and Z, and in red for lead VM. For leads X, Y, and Z, the individual beats that were aligned to construct the X, Y, and Z median beats are shown with black dotted lines. There are no individual beats shown for the VM median beat because the median VM beat (VM_{med}) is created only with the median X, Y, and Z beats as $VM_{med} = \sqrt{X_{med}{}^2 + Y_{med}{}^2 + Z_{med}{}^2}$. The location of the * corresponds the location of the maximum T wave amplitude in the VM lead. When a beat is selected in the '**Annotated Beat Viewer**' (GUI item 8.1), the selected beat is highlighted in green:

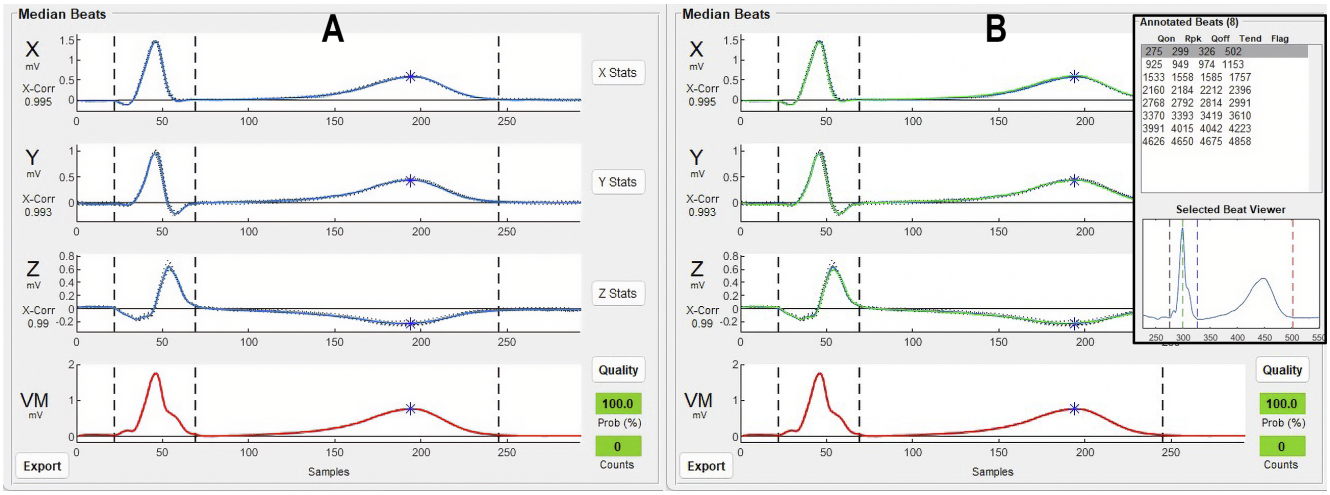


Figure 9: **A:** When a beat is not selected the individual beats that make up each of the X, Y, and Z median beats are shown in black dashed lines. **B:** When a beat is selected in the '**Annotated Beats**' section (GUI item **8.1**), the selected beat in the X, Y, and Z leads is highlighted in green.

---

2. Mean normalized cross correlation (NCC/'X-Corr') for the X, Y, and Z beats, respectively. Values close to 1.0 indicate that all beats in that lead which were used to construct the median beat were very similar in morphology and alignment, while low values suggest that there are multiple beat morphologies that differ significantly from each other and/or poor beat alignment which might impact the quality of median beat construction and subsequent calculations. This is used primarily as a quality control, and the lowest of the normalized cross correlation values for the X, Y, and Z leads is used as part of the quality assessment after ECG processing (see **Chapter 16**).

---

3. The X Stats, Y Stats, and Z Stats buttons display information on the individual beats used to create the median beat in the X, Y, and Z leads, respectively. This information can be useful to assess for outlier beats or beats that are causing issues with median beat construction. An example of the figure produced when clicking these buttons is shown in **Figure 63**.

---

4. The **Quality** button opens a figure that shows various quality metrics for the processed ECG. Further information can be found in **Chapter 16** and **Figure 44**.

---

5. Measures of ECG processing/annotation quality (see **Chapter 16**). The 'Prob (%)' value is a predicted probability between 0 and 1 (expressed as a percentage) that is the output of a logistic regression which takes various morphologic features of the processed median beat as input and outputs the probability of the processed median beat being "good" quality.

Values above a set cutoff in `quality_presets.csv` are shown with a **green** background, while values below this cutoff are shown in **red** background. If the value is 'NaN' this signifies a significant issue with annotation that should be investigated (or an ECG has not been processed). See **Chapter 16** for further details.

The 'Counts' value reports the number of quality parameters that are outside of the bounds set in `quality_presets.csv`. Values of 0 are shown in **green**, while values > 0 are shown in **red**. The figure opened by clicking on the **Quality** button (see **Figure 44**), displays the individual quality parameters that are used for this metric. See **Chapter 16** for details.

---

6. The Export button exports the 16 median beats and median beat annotations as `.mat` and `.csv` files. The exported files will appear in the same directory as the ECG file with the naming convention `<filename>_xyz_medians.mat` and `<filename>_xyz_medians.csv`. The `.mat` file contains a structure `medians` that contains fields corresponding to all 16 leads (12 leads, X, Y, Z, and VM) (eg `medians.V1` loads the lead V1 median beat). The `.csv` file contains 16 columns corresponding to the 16 leads in the order I, II, III, avR, avF, avL, V1, V2, V3, V4, V5, V6, X, Y, Z, and VM. In both file formats, exported signals are in units of mV. In the `.mat` file, the median beat annotations are saved as `median.Q` (Q_{on}), `median.S` (Q_{off}), and `median.Tend` (T_{off}) in samples. Median beat annotations are not saved to the `.csv` file.

12.6 GUI Section 5 – Select Displayed Information

Section 5:

Select what information is shown in the GUI information display (GUI Section 6).

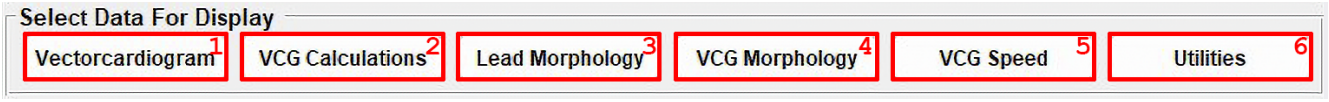


Figure 10: GUI section 5.

1. The **Vectorcardiogram** button displays the 3D VCG. See GUI Section 6.1 in **Chapter 12.7** for further details.

---

2. The **VCG Calculations** button displays a set of measurements obtained from the median ECG/VCG, and the majority of results contained in `VCG_Calc` results class. See GUI Section 6.2 in **Chapter 12.7** and **Chapter 24** for further details.

---

3. The **Lead Morphology** button displays a pop-out window with all 16 median beats, fiducial points, and the measurements of the R, S, and T waves. The QRS axis is also displayed. This displays the majority of results contained in the `Lead_Morphology` results class. See GUI Section 6.3 in **Chapter 12.7** and **Chapter 24** for further details.

---

4. The **VCG Morphology** button displays information on the morphology of the QRS and T loops, including coplanarity, total cosine R-to-T (TCRT), roundness, area, and perimeter, including the majority of results contained in the `VCG_Morphology` results class. See GUI Section 6.4 in **Chapter 12.7** and **Chapter 24** for further details.

---

5. The **VCG Speed** button displays measurements of median VCG speed and allows graphical display of median VCG speed and acceleration. See GUI section 6.5 in **Chapter 12.7** for further details.

---

6. The **Utilities** button brings up a set of ECG/VCG utilities and allows additional control over GUI function and output. See GUI Section 6.6 in **Chapter 12.7** further details.

12.7 GUI Section 6 – GUI Information Display

Section 6.1:

The **Vectorcardiogram** button displays the median beat VCG with various visualization options. The figure can be rotated in 3D.

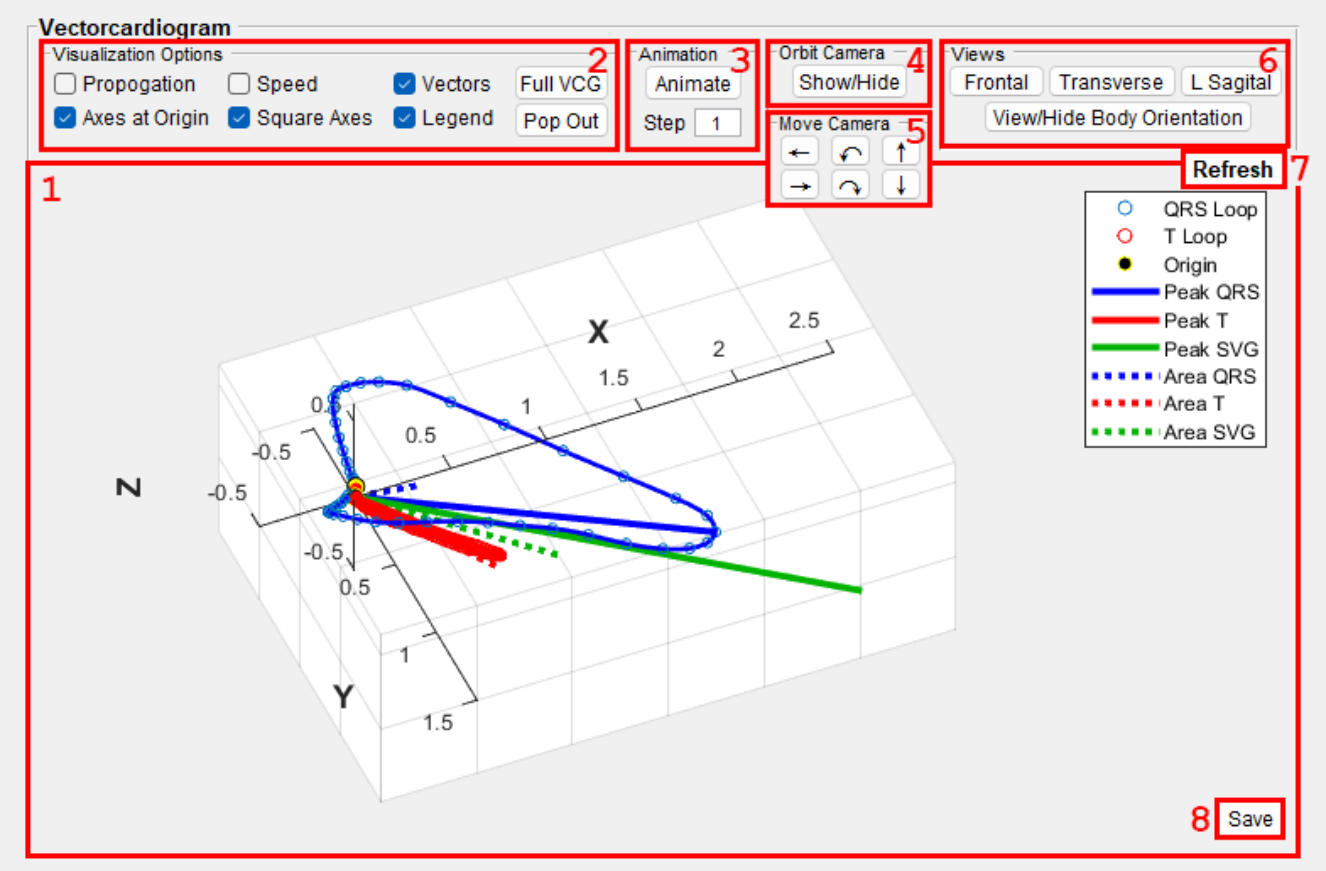


Figure 11: GUI Section 6.1

1. The median beat VCG is shown in the '**Vectorcardiogram**' section of the GUI. The QRS loop is shown in blue and the T wave loop is shown in red. 'Visualization Options' (GUI item **6.1.2**) contains options for visualizing the VCG. When new options are checked/unchecked, clicking **Refresh** (GUI item **6.1.7**) will update the VCG graph without recalculating values. The VCG can be rotated with the mouse by using the rotation tool (GUI item **10.2**) or the 'Orbit Camera' in GUI item **6.1.4**.

2. VCG Visualization Options:
   - ☑ Propogation : When checked this option shows a set of green circles at the start of the QRS loop and a set of yellow circles at the start of the T wave loop. This can be useful to visually determine the direction of rotation around each VCG loop.
   - ☑ Speed : When checked, this option shows the instantaneous speed (mV/ms) of the VCG loop. The loop coloring changes to reflect the relative speed according to the color key that is displayed. Faster sections of the loops are shown in red, and slower sections of the loops are shown in blue (see GUI section 6.5 in **Chapter 12.7**).
   - ☑ Vectors : When checked, this option shows the peak and area QRS (blue), T wave (red), and QRST/SVG (green) vectors. Peak vectors are shown as solid lines, while area vectors are shown as dashed lines.
   - ☑ Axes at Origin : When checked, this option moves axes labeling inside the 3D figure as opposed to being on the outside of the figure.
   - ☑ Square Axes : When checked, this option makes the X, Y, and Z axes all have the same scaling. If this is not checked the loop morphology may be distorted, but it may be easier to visualize some parts of the VCG loops.
   - ☑ Legend : Turns on or off the figure legend on the right hand side of the figure.
   - Full VCG : Clicking this button shows the plot of the full VCG throughout the entire VCG tracing, not just the median beat (see **Figure 64**).
   - Pop Out : Clicking this button allows visualization/manipulation of the VCG in a new, larger window.

---

3. The Animate button opens an animated figure of the VCG loop. The animation steps are in samples set by the 'Step' textbox.

---

4. The Orbit Camera opens a second toolbar that can be used to manipulate the camera for the 3D VCG. See `https://www.mathworks.com/help/matlab/creating_plots/view-control-with-the-camera-toolbar.html` for additional details.

---

5. This section is used to rotate the 3D VCG without using the rotation tool (GUI item 10.2). All movements are relative to the user.
   - ← Rotates the VCG towards the **left** of the user.
   - → Rotates the VCG towards the **right** of the user.
   - ↶ Rotates the VCG **counter–clockwise** relative to the user.
   - ↷ Rotates the VCG **clockwise** relative to the user.
   - ↑ Rotates the VCG **cranial** relative to the user.
   - ↓ Rotates the VCG **caudal** relative to the user.

6. This section is used to rotate the 3D VCG to common viewing angles.
   - Frontal Displays the VCG in the XY plane.
   - Transverse Displays the VCG in the XZ plane.
   - L Sagital Displays the VCG in the YZ plane.
   - View/Hide Body Orientation Shows a head/face to orient the VCG loops in 3D space. The rotation of the VCG and this face are linked, so rotating one will also rotate the other by the same amount and to the same viewing angle.

---

7. The **Refresh** button reloads the VCG with the active VCG visualization options set in GUI item 6.1.1 above.

---

8. The Save button saves the current VCG as a `.png` image.

Section 6.2:

The **VCG Calculations** button displays results from the `VCG_Calc` results class and sets options for data export.

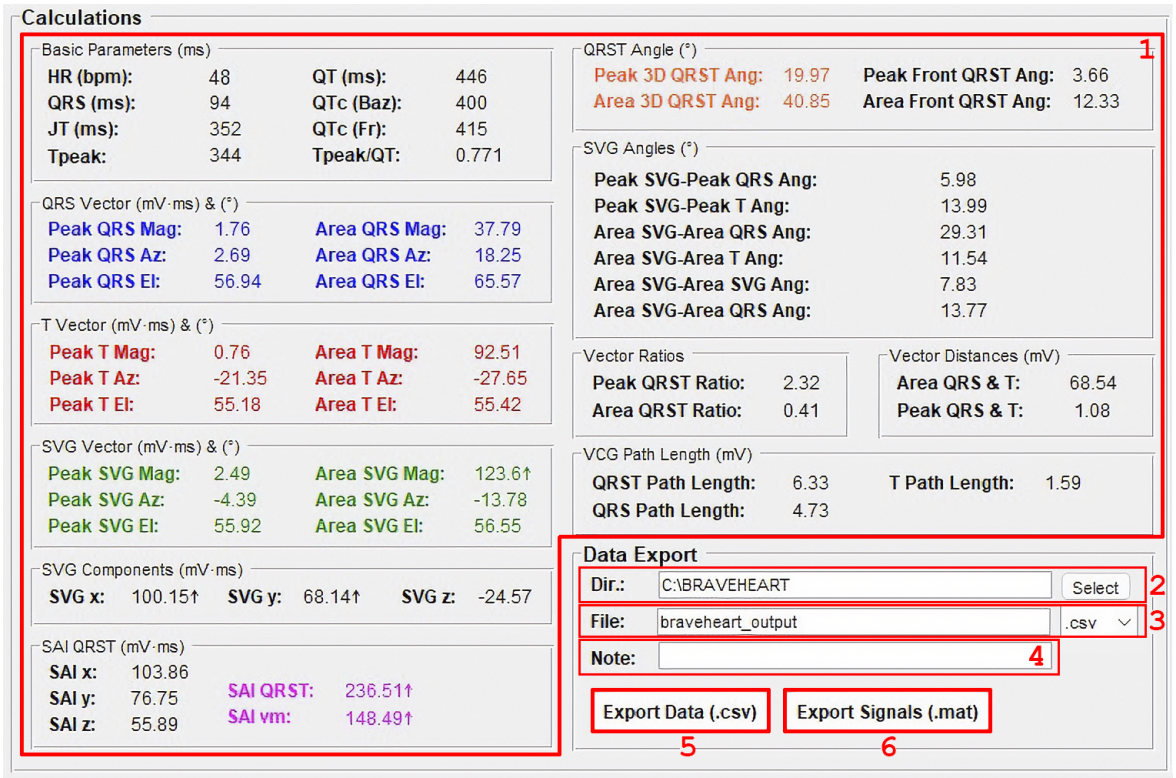


Figure 12: GUI Section 6.2

1. Displays values of various ECG/VCG calculations from the `VCG_Calc.m` results class based on the median beat. If ☑ Show Normal Ranges is checked in the **Utilities** section (GUI item **6.6.6**) then for select values, if they are out of range, to the right of the value there will be an arrow reflecting if the value is above (↑) or below (↓) the normal range (see **Figure 55** and **Chapter 22**). Definitions for azimuth and elevation can be found in **Figure 71**.

---

2. Textbox that shows the working directory for saving files. All saved files are stored in the directory displayed here. When an ECG is loaded, the working directory for saving data is set to the directory which contains the ECG file that was loaded. To change the working directory, click on the Select button and choose a new directory.

---

3. Textbox that specifies the name of the file used to export calculations/data from the current ECG/VCG. The dropdown allows the user to choose `.csv` or `.xlsx` format.

---

4. Textbox that allows the user to specify a note to be appended to the export file.

---

5. The **Export Data** button exports all of the current data from the active ECG/VCG (including all calculations from all 3 results classes, Annoparameters, and beat annotations), to a text file with the file name and extension as specified in GUI item **6.2.3**. The button text changes to reflect the value in the file extension dropdown box (`.csv` or `.xlsx`).

   If a file with the same name/location as specified in the saving directory (GUI item **6.2.2**) and filename (GUI item **6.2.3**) does not exist, a new file is created. If a file with the same name/location as specified already exists, the current data is appended to the next empty row in the existing file. In this way the user can create a singular file with the results from multiple ECGs, or results from the same ECG with different processing parameters/included beats.

   When the export is successful, **Success!** appears to the right of the buttons.

---

6. The Export Signals (.mat) button exports the signal data (ECG/VCG signals, median beat signals, individual beats) and results in a `.mat` format file which contains a data structure with all of the exported data that can be analyzed outside of BRAVEHEART. Details on the contents of this data structure can be found in **Chapter 25**.

Section 6.3:

The **Lead Morphology** button opens a figure which contains the median beat R, S, and T wave measurements (all units in mV) for all 16 leads. The information displayed here is from the `Lead_Morphology.m` results class. The QRS frontal axis is also displayed.

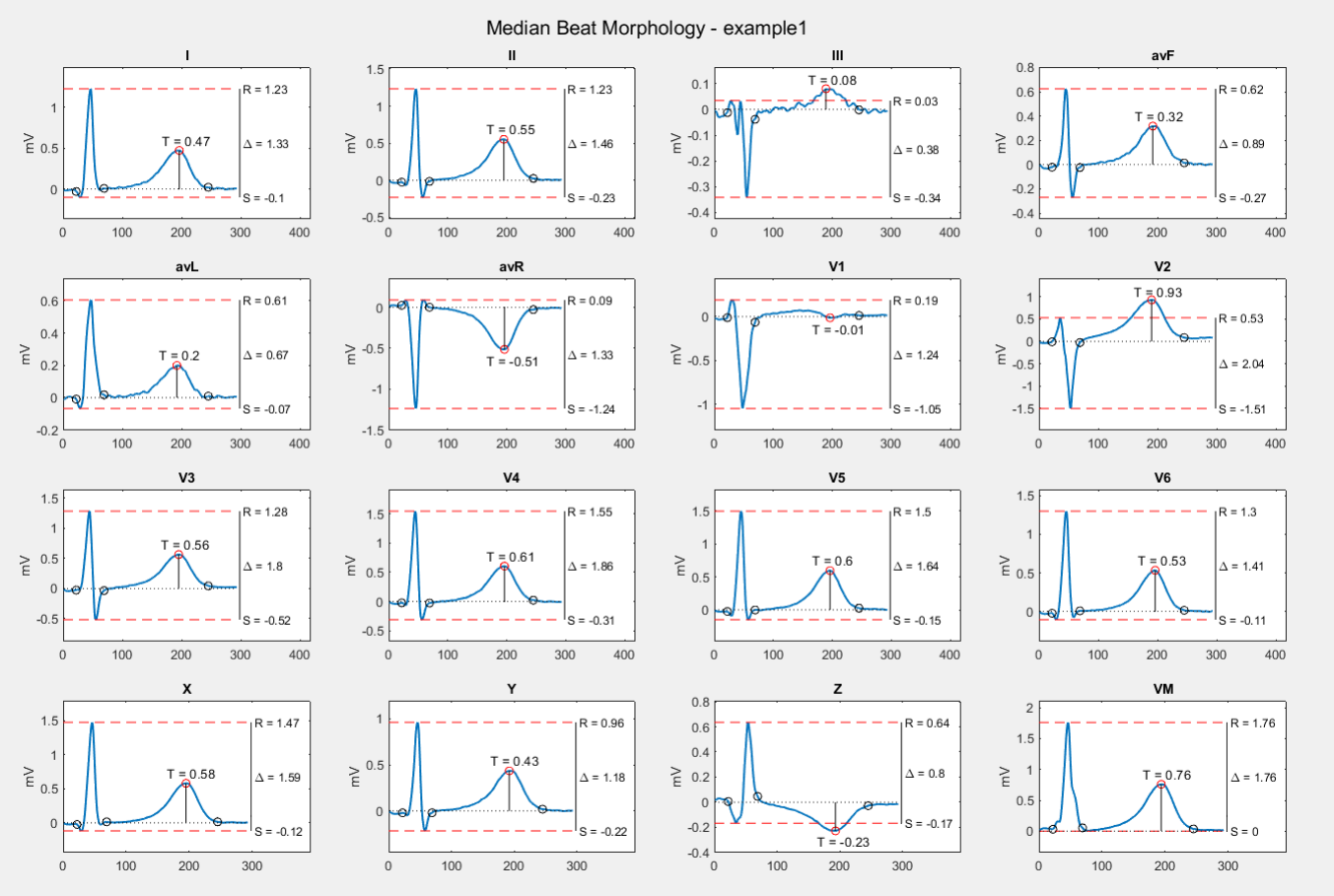


Figure 13: Figure displayed when the Lead Morphology button is pressed. The figure displays the R, S, and T wave measurements for all 16 leads. All values are in units of mV.

Section 6.4:

The VCG Morphology button displays information on VCG loop morphology and associated parameters. The information displayed here is contained in the `VCG_Morphology.m` results class.

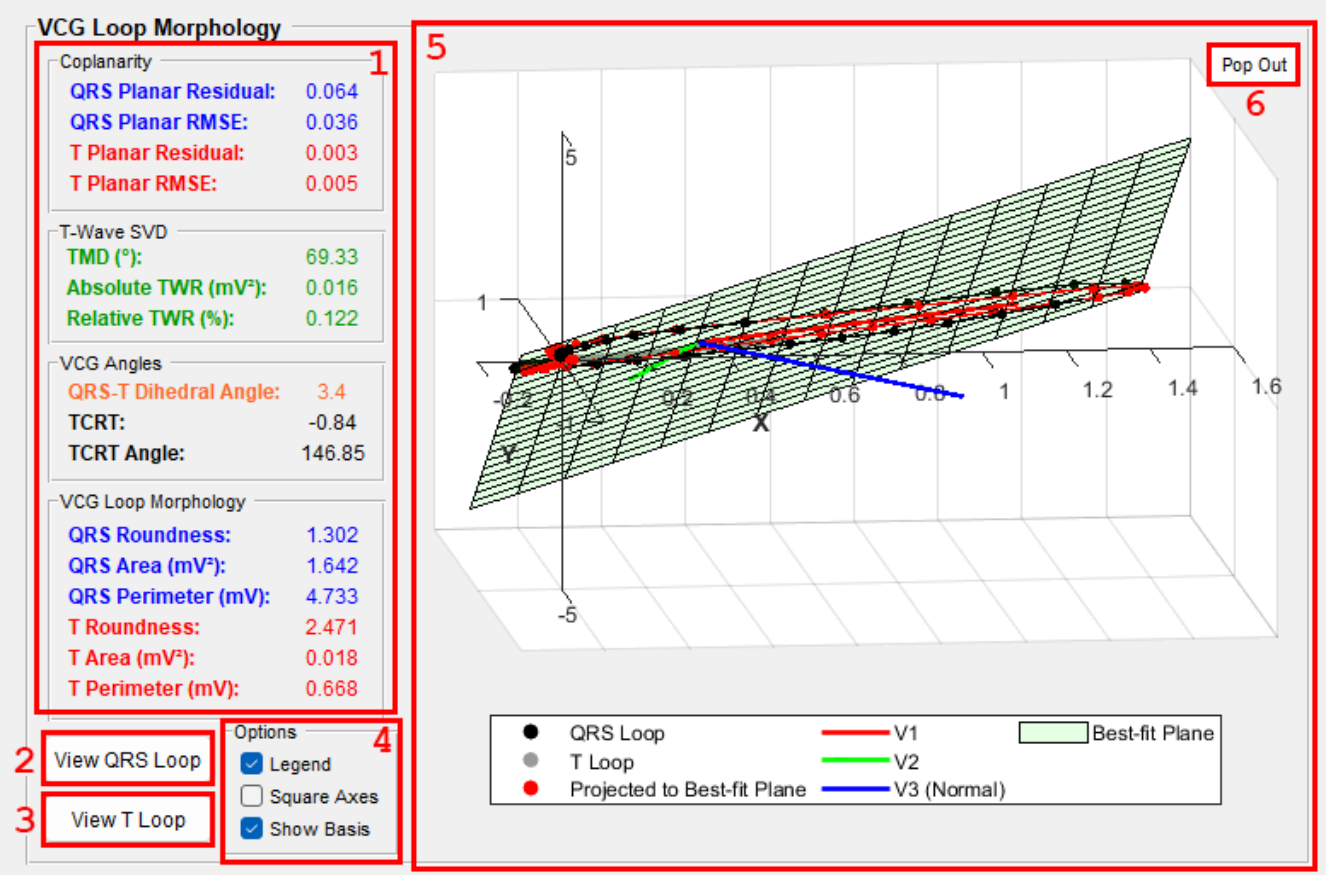


Figure 14: GUI Section 6.4

1. Displays values of various VCG morphology calculations such as coplanarity (how planar the QRS and T loops are), Total Cosine R-T, T wave mechanical dispersion, T wave residuum, and QRS and T wave loop roundness and area which are part of the `VCG_Morphology` results class. All variables calculated in the `VCG_Morphology` results class are not displayed here. Further details can be found in the methods manuscript [1] and in **Chapter 24**.

---

2. The View QRS Loop button shows the best fit plane for the QRS loop and projections of the loop into the best fit plane. Vectors that span the best fit plane can be displayed as noted in GUI item 6.4.4. The figure can be rotated.

---

3. The View T Loop button shows the best fit plane for the T loop and projections of the loop into the best fit plane. Vectors that span the best fit plane can be displayed as noted

in GUI item 6.4.4. The figure can be rotated.

---

4. These checkboxes control the appearance of the displayed QRS or T loop figure:
   - ☑ Legend : shows the figure legend
   - ☑ Square Axes : makes the X, Y, and Z axes have equal spacing.
   - ☑ Show Basis : shows the orthogonal basis vectors for the best fit plane

---

Section 6.5:

The VCG Speed button displays information on VCG loop speed.

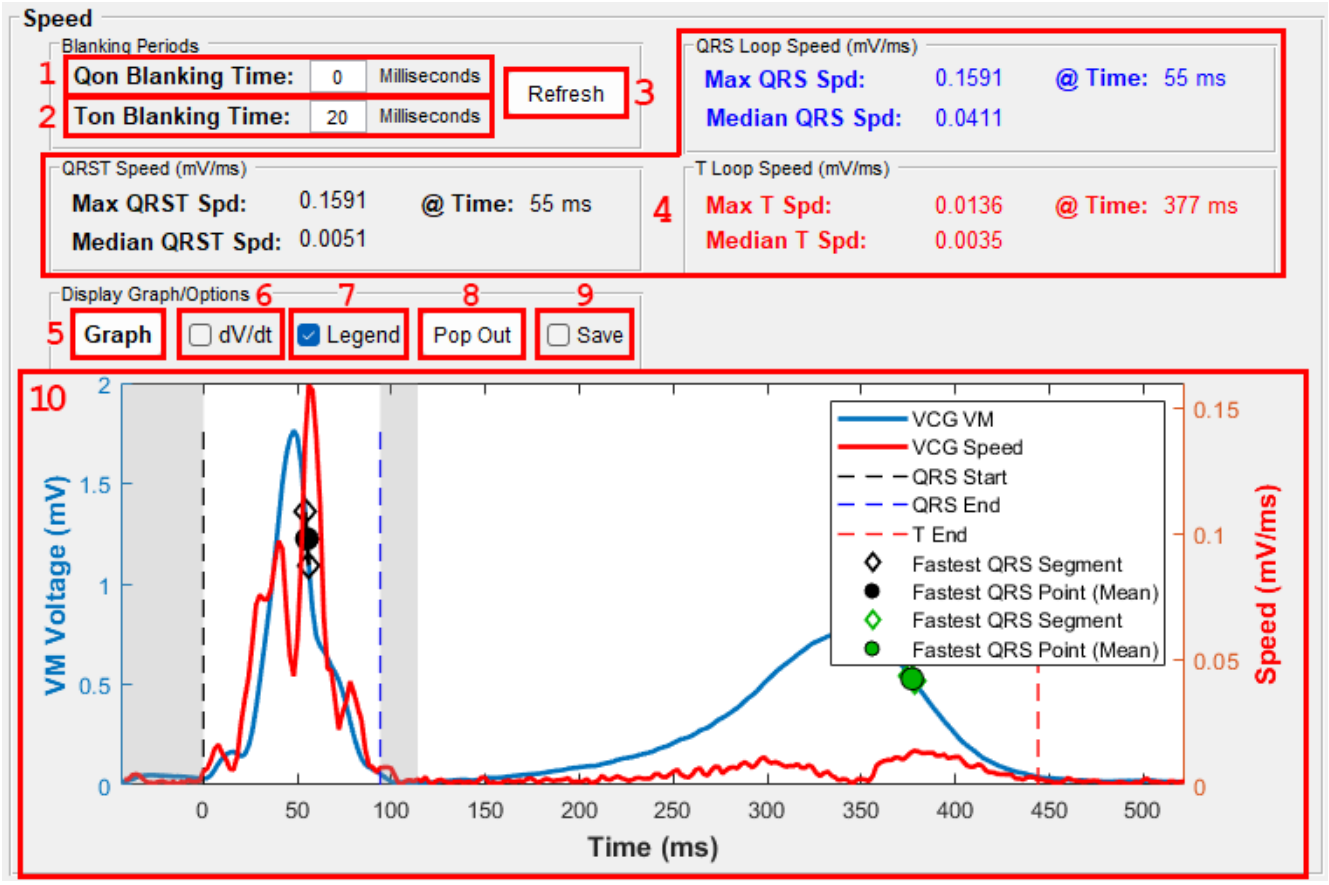


Figure 15: GUI Section 6.5

1. Textbox to set the value of `blanking_window_q` in `Annoparams.m`. This corresponds to the length of the gray shaded area in milliseconds after QRS onset (black vertical dashed line) in the figure above (GUI item 6.5.10). The gray shaded area before QRS onset is automatically blanked from all calculations. Any part of the QRST signal within this gray shading is ignored when calculating maximum speed or maximum speed location. This can be useful to isolate specific parts of the QRST complex, or to deal with large/wide pacing

artifact that contaminates the initial part of the QRS complex. The nominal value for `blanking_window_q = 0`. See **Chapter 6.2** for additional details.

---

2. Textbox to set the value of `blanking_window_t` in `Annoparams.m`. This corresponds to the length of the gray shadded area in milliseconds after the end of the QRS complex (blue vertical dashed line) in the figure above (GUI item **6.5.10**). Any part of the T wave within this gray shading is ignored when calculating the T wave maximum speed or maximum speed location. This blanking period had no effect on speed calculations involving the entire QRST complex. Setting `blanking_window_t` > 0 is useful in cases where the annotation of QRS offset is a few samples too early, as in this case, part of the end of the QRS complex which can have significantly higher speed than the T wave, might be considered as the start of the T wave. The nominal value for `blanking_window_t = 20`. See **Chapter 6.2** for additional details.

---

3. The Refresh button updates the speed values in GUI item **6.5.4** (see below) based on the value set in the 'Blanking Time' textboxs (GUI items **6.5.1** and **6.5.2** which set the values of `blanking_window_q` and `blanking_window_q`, respectively, in `Annoparams.m`. Use the Refresh button to recalculate values of speed using new values of `blanking_window_q` and `blanking_window_q` and update the Speed figure (GUI item **6.5.10**) automatically.

---

4. Values for speed calculations of the entire QRST complex, QRS complex alone, and T wave alone.

---

5. The **Graph** button displays the VCG speed figure in GUI item **6.5.10** based on the values of `blanking_window_q` set in GUI item **6.5.1**, `blanking_window_t` set in GUI item **6.5.2**, the ☐ dV/dt checkbox set in GUI item **6.5.6**, and the ☑ Legend checkbox set in in GUI item **6.5.7**.

---

6. The ☐ dV/dt checkbox sets if acceleration is also shown in the speed figure (GUI item **6.5.10**). Currently, the values and locations of maximum acceleration are not included in the BRAVEHEART output files.

---

7. The ☑ Legend checkbox sets if the figure legend is displayed. If the legend is obstructing view of the T wave, the legend can be disabled.

---

8. The Popout button opens the speed figure displayed in GUI item **6.5.10** in a new window to more easily adjust the figure. When the ☐ Save checkbox (GUI item **6.5.9** below) is also checked, the Popout button also saves the speed figure.

9. The ☐ Save checkbox saves the speed figure shown in GUI item **6.5.10** as a `.png` file named `<filename>_speed.png` when the **Popout** button (GUI item **6.5.8**) is clicked.

10. Figure showing speed (and acceleration if ☑ dV/dt is checked). The blanking periods, which are assigned by GUI items **6.1.1** and **6.1.2** above are shown as gray shading after QRS onset (black vertical dashed line) and QRS offset/T wave onset (blue vertical dashed line). The fiducial points for Q_{on}, Q_{off}, and T_{off} are shown with colored, dashed lines. The locations of the fastest segment of the QRS complex and T wave are noted with shaded circles as shown in the legend. If the ☑ dV/dt checkbox (GUI item **6.1.6**) is checked, the acceleration is also displayed. Of note, the X-axis on this figure is is milliseconds after QRS onset (Q_{on}).

Section 6.6:

The **Utilities** button brings up BRAVEHEART utilities and options which can be used to control ECG processing and utilize other useful functions and links.

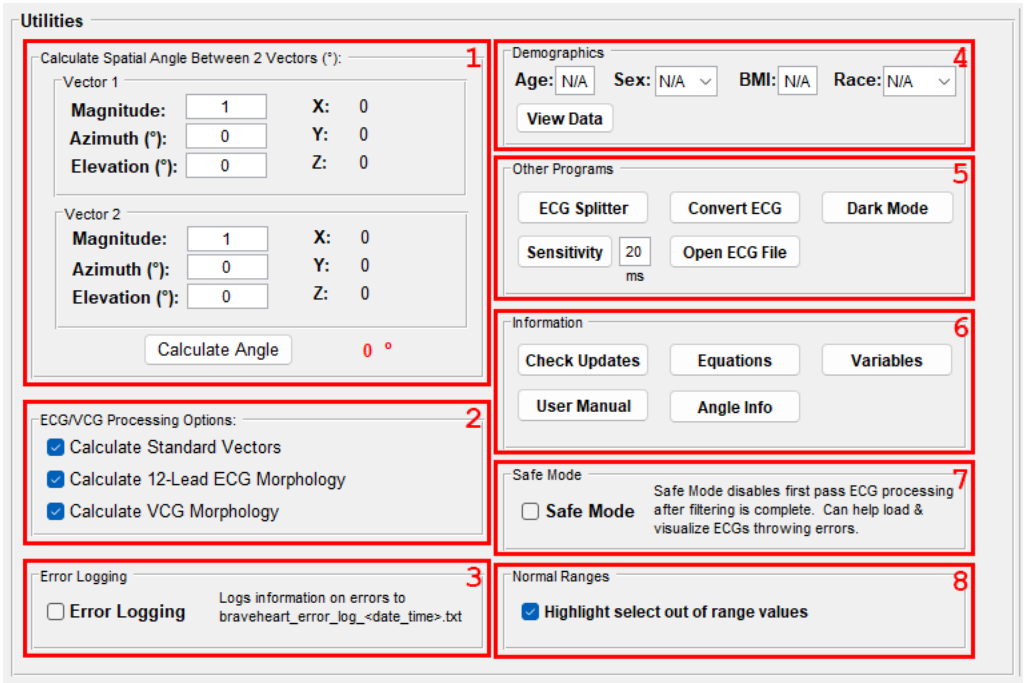


Figure 16: GUI Section 6.6

1. If needed, BRAVEHEART can convert vectors defined by magnitude, azimuth, and elevation (see **Figure 71**) into Cartesian coordinates. If 2 vectors are entered, the 3D angle between them can also be calculated by clicking the Calculate Angle button.

2. By default, BRAVEHEART calculates parameters in all 3 result classes (see **Chapter 24**). In certain cases the user may only care about some of the results classes. The 3 checkboxes here allow the user to enable/disable processing of specific results classes to speed up batch processing of large volumes of ECGs if the data contained in one or more specific results classes is not needed:
   - ☑ Calculate Standard Vectors enables/disables the `VCG_Calc` results.
   - ☑ Calculate 12-Lead ECG Morphology enables/disables the `Lead_Morphology` results.
   - ☑ Calculate VCG Morphology enables/disables the `VCG_Morphology` results.

   These checkboxes set the values of `vcg_calc_flag`, `lead_morph_flag`, and `vcg_morph_flag` (see **Chapter 13**).

3. The ☐ **Error Logging** checkbox enables error logging to assist with troubleshooting ECGs that will not process correctly. Further details can be found in **Chapter 31**.

4. BRAVEHEART includes data on estimated normal limts for a selection of VCG parameters based on age, sex, body mass index (BMI), race, and heart rate (see **Chapter 22**). This section of textboxes and dropdowns allows the user to specify the age, sex, BMI, and race (white or not white) to generate expected normal limits for these select parameters.

   ECG files (usually `.xml` or `.dcm`) that contain information on patient age, sex, and race can have this information automatically pulled from the ECG file and extracted to this location of the GUI. If data on age, sex, and/or race is not provided in the ECG, the data can be manually input via the text boxes and dropdowns in this section. To obtain the most accurate normal ranges for the ECG being processed, the BMI can be entered here as well (this currently is not pulled from ECG files because it is not usually provided in the metadata). Values of "N/A" mean that the data was not provided, and missing data is dealt with as described in **Chapter 22**.

   Clicking on the **View Data** button will parse the ECG file for demographic information and then display it in a chart:

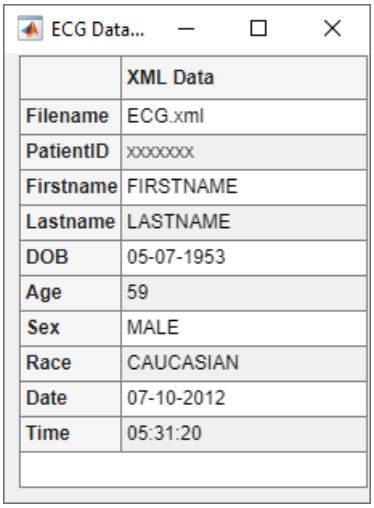


Figure 17: Information parsed from the ECG file using the **View Data** button

Values that are missing will display as "N/A".

If the **☑ Highlight select of of range values** (GUI item **6.6.8** below) is checked, values outside of the normal ranges determined by the demographics entered in this section are noted on the GUI results displayed in the '**VCG Calculations**' section of the GUI (GUI item **6.2.1**) with ↑ or ↓ reflecting if the value is above or below the normal range, respectively. Further details of displaying values outside of the normal range can be found in the documentation for GUI item **6.6.8** and **Chapter 22**.

---

5. This section is used to open various utility programs that may be useful to the user:

   In certain situations the user may want to split an ECG file into 2 separate files to remove an area of significant noise/artifact, or to isolate a specific finding. Clicking on **ECG Splitter** allows the user to split the current file into 2 separate files by specifying a sample at which to split the ECG. In a future release we plan to allow the user to maintain the current file format when splitting ECGs, but in the current version of BRAVEHEART all split ECGs are in the `'unformatted'` format which encodes the 12 leads as 12 columns in units of mV (see **Chapter 23**).

   The current ECG can be converted into a different format using the **Convert ECG** button. This function is currently only able to convert into `'unformatted'` ECGs, although additional formats will be added in the future. When you click on the **Convert ECG** button a pop-up asks if you want to convert the currently loaded ECG or a directory of ECGs which used the `batch_convert.m` function. This function can be called outside of the BRAVEHEART GUI if needed (see source code of `batch_convert.m` for

details on calling the function outside of the GUI).

The **Sensitivity** button performs sensitivity analysis on the current ECG to determine how much changes in fiducial point annotations affect the reported measurements. The textbox next to the **Sensitivity** button is used to specify an interval in ms by which the Q_{on}, Q_{off}, and T_{off} fiducial points are adjusted, and all results recalculated. For example, if the textbox is set to the nominal value of '20' ms, clicking the **Sensitivity** button will calculate results with all ranges of Q_{on} from Q_{on} - 20 ms to Q_{on} + 20 ms, and similarly for Q_{off} ±20 ms and T_{off} ±20 ms. Results are saved to the working directory with the filename `<filename>_sensitivity_analysis.csv`.

The currently loaded ECG file can be directly opened using whatever program is associated with the ECG file's file extension using the **Open ECG File** button.

The Dark Mode or Light Mode buttons switch between light/dark GUI color schemes. See **Chapter 28** for further details.

---

6. This section can be used to display and obtain various useful information:
   - The **Check Updates** button opens the BRAVEHEART GitHub website, `http://www.github.com/BRAVEHEART`, so the user can check for any software updates.
   - The **Equations** button displays various equations used for calculations.
   - The **User Manual** button displays this document.
   - The **Angle Info** button displays a figure explaining the conventions used for angles in BRAVEHEART calculations (see **Figure 71**).
   - The **Variables** button displays the variable names and descriptions for output files.

---

7. Occasionally ECGs will throw errors when they are being loaded into BRAVEHEART. When checked, the ☐ **Safe Mode** checkbox disables first pass ECG processing and annotation after initial filtering is complete. This can help trouble shoot ECGs that throw errors. The ECGs can be viewed by clicking on the 12L ECG and other buttons in the **'View/Save ECG'** section (GUI section 7). Further details of Safe Mode can be found in **Chapter 31**.

---

8. The ☑ **Highlight select out of range values** checkbox displays marks next to select measurements that are out of range in the '**VCG Calculations**' section (GUI item **6.2.1**). Values that are above the normal range are displayed with an ↑, and values that are below the normal range and displayed with a ↓. If ☐ **Highlight select out of range values** is not checked, the arrows to visualize out of range values are not displayed. Further details of

which parameters have normal ranges calculated are available in **Chapter 22**.

12.8 GUI Section 7 – Display Various Figures

Section 7:

View/Save ECGs, VCGs, and other figures.

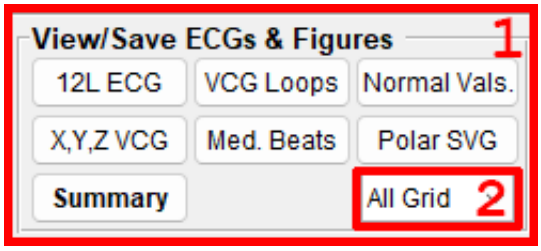


Figure 18: GUI Section 7

This section of the GUI is used to display and save various ECG/VCG figures and visualizations. To save a figure, click the Save .png button once the figure opens.

1. Generate various figures of ECGs, VCGs, and other visualizations:
   - The 12L ECG button displays the 12-lead ECG as 12 continuous rhythm strips. The 12-lead ECG can be displayed after the ECG is loaded and before the **Calculate** button has been pressed. An example 12-lead ECG figure is shown in **Figure 52**.
   - The X, Y, Z VCG button displays continuous rhythm strips of the X, Y, Z, and VM leads. The VCG ECG can be displayed after the ECG is loaded and before the **Calculate** button has been pressed. An example VCG figure is shown in **Figure 53**.
   - The Med. Beats button displays all 16 median beats. This only works if the **Calculate** button has been pressed and the median ECG has been processed. An example median beats figure is shown in **Figure 54**.
   - The **Summary** button displays measurements of the summary figure (See **Figure 50** and **Figure 51**). This only works if the **Calculate** button has been pressed and the median ECG has been processed.
   - The Polar SVG button brings up a plot that shows orientation of the spatial ventricular gradient (SVG) and QRST angles. This only works if the **Calculate** button has been pressed and the median ECG has been processed.
   - The Normal Vals. button brings up a figure that shows age, gender, race, and body mass index specific values for various VCG parameters. An example figure is shown in **Figure 55**. See **Chapter 22** for further details.
   - The VCG Loops button brings up a figure that shows the median X, Y, and Z beats and the corresponding VCG loops. This only works if the **Calculate** button has been

pressed/median ECG has been processed. An example figure is shown in **Figure 56**.

2. Dropdown to choose what gridlines are displayed when using 12L ECG, X, Y, Z VCG, or Med. Beats. 'All Grid' enables major and minor grids (large and small boxes), 'Maj Grid' enables only the major grid (large boxes), 'Min Grid' enables only the minor grid lines (small boxes), and 'No Grid' disables the grid completely.

12.9 GUI Section 8 – Beat Editing, Addition, and Removal

Section 8:

Edit the fiducial point annotations of individual beats, remove unwanted beats, or manually add new beats. After editing, adding, removing beats, a new median beat is created and calculations are performed on the new median beat.

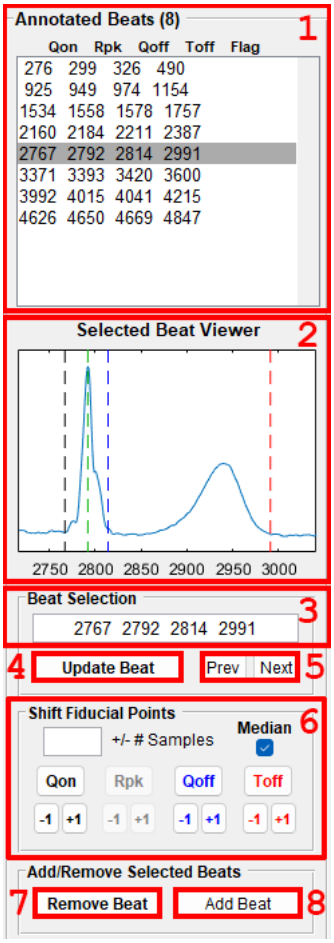


Figure 19: GUI Section 8.

1. The '**Annotated Beats**' window lists the annotations for all individual beats that are included in the median beat. The order of anotations is shown at the top of the box as Q_{on}, R peak, Q_{off}, and T_{off} (in samples). Beats that are detected as PVCs/non-dominant beats will have a '#' next to their annotations. Beats that are detected as outliers will have a '**' next to their annotations. Some beats may be detected as both a a PVC and an outlier, and will have both symbols. Clicking on a beat in the '**Annotated Beats**' window displays the beat and fiducial points in the '**Selected Beat Viewer**' (GUI item 7.2 below), highlights it in green in the **Median Beat Viewer** (GUI item 4.1) and

highlights the individual beat with a green * in the '**X-, Y-, Z-, Vector Magnitude Leads**' section (GUI item **2.1**).

---

2. The '**Selected Beat Viewer**' shows a zoomed in view of the beat that is selected in the '**Annotated Beats**' window along with its annotations visualized as a black dashed line (Q_{on}), green dashed line (R peak) , blue dashed line (Q_{off}) , and red dashed line (T_{off}). These lines can be dragged to adjust the respective fiducial point locations. The sample values that correspond to the location of the 4 colored dashed lines are reflected in the '**Beat Selection**' textbox below the '**Selected Beat Viewer**' (GUI item **7.3**). As the lines are moved, the values in the '**Beat Selection**' textbox will reflect the new location of the fiducial points (see **Chapter 5.6**). You must click the **Update Beat** (GUI item **7.4**) to update the selected beat fiducial points when adjusting an individual beat fiducial point values.

---

3. The '**Beat Selection**' textbox updates with the selected beat's fiducial points in the order of Q_{on},Rpeak , Q_{off}, or T_{off} as they are adjusted. The fiducial points can be edited by dragging the colored dashed lines that correspond with Q_{on}, R peak, Q_{off}, and T_{off} as noted above, manually by updating their values in the textbox, or by shifting values by ± 1 sample using the **−1** and **+1** buttons corresponding to Q_{on}, Rpeak , Q_{off}, or T_{off} in GUI section **7.6**. See **Chapter 5.6** for additional information.

Once the adjusted fiducial points are in the '**Beat Selection**' textbox, clicking the **Update Beat** button (GUI item **7.5**) or **Remove Beat** button (GUI item **7.8**) will update the fiducial points or remove the beat entirely, respectively.

---

4. The **Update Beat** button takes the 4 values in the '**Beat Selection**' textbox corresponding to Q_{on}, R peak, Q_{off}, and T_{off} (in that order), saves these new values for the selected beat, and then recalculates the median beat and associated measurements using the updated beat fiducial points.

---

5. The Prev and Next buttons can be used to cycle through beats in the '**Annotated Beats**' listbox without clicking on them.

---

6. The '**Shift Fiducial Points**' textbox allows manual adjustment of fiducial points for individual beats or the median beat. All adjustments are in **samples**, not ms. The main behavior of the buttons in this section is dictated by if ☑ Median is checked or not.

If ☑ Median is checked, adjustments are applied to the current median beat. Note that if

median beat adjustment is active, the R peak cannot be adjusted. There are 2 ways of adjusting the median beat Q_{on}, Q_{off}, or T_{off}. First, you can adjust a fiducial point by ± 1 sample by clicking on the [−1] and [+1] buttons below the relevant fiducial point. For example, to shift Q_{on} by $+1$ sample, click on the [+1] button below [Qon], and to shift T_{off} by -1 sample, click on the [−1] button below [Toff]. The fiducial points of the median beat are adjusted and calculations are updated.

If larger adjustments are needed, placing an integer n in the textbox and clicking on one of the buttons ([Qon], [Qoff], or [Toff]), will shift the relevant median beat fidicual point by n samples. For example, to shift the median beat T_{off} later by 3 samples, enter '3' in the textbox and click the [Toff] button. To shift the median beat Q_{on} earlier by 2 samples enter '-2' in the textbox and click the [Qon] button. The fiducial points of the median beat are adjusted and calculations are updated.

See **Chapter 5.11** for an example of median beat annotation adjustment.

If ☐ Median is unchecked, placing an integer n in the textbox and clicking on one of the buttons ([Qon], [Rpk], [Qoff], or [Toff]), will shift the relevant fiducial point of ALL individual beats by n samples. In practice, this is useful if the median beat is being truncated before the end of the T wave which is preventing annotation of the median beat T_{off}; shifting T_{off} (click [Toff]) by n samples in all individual beats has the effect of extending the window for median beat creation by n samples. The window before Q_{on} can be similarly extended by using a negative value and clicking [Qon].

If ☐ Median is unchecked, the [−1] and [+1] buttons only work if a beat is selected and displayed in the '**Selected Beat Viewer**'. If a beat is selected, the [−1] and [+1] buttons shift the relevant fiducial point by ± 1 sample, and update the value in the '**Beat Selection**' textbox. You must click the [Update Beat] (GUI item **7.4**) to update the selected beat fiducial points when adjusting an individual beat fiducial point values.

Note that unlike with median beat fiducial point adjustment, R peak locations **can** be adjusted on individual beats (one at a time or as a group)

---

7. The **Remove Beat** button removes the selected beat from the analysis and then recalculates the median beat and associated measurements. See the quick start guide (**Chapter 5.7**) for an example.

8. The Add Beat button takes the values in the '**Beat Selection**' textbox, creates as new beat using these fiducial points, and then recalculates the median beat and associated measurements. In general, there are few uses for this function, as beats should be automatically detected, but in rare cases where a beat needs to be added manually, it can be done this way.

12.10 GUI Section 9 – PVC/Non-Dominant and Outlier Beat Removal

Section 9:

Analyze and remove PVCs/non-dominant beats and outlier beats. Detailed descriptions of PVC and outer identification/removal can be found in **Chapter 15.4** and **Chapter 15.5**.

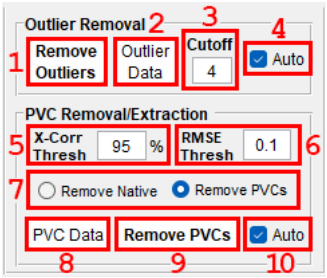


Figure 20: GUI Section 9

1. The **Remove Outliers** button removes all beats that have been flagged as outliers and then recalculates the median beat and associated measurements. Further details can be see in quick start Example 3 (**Chapter 5.3**), **Chapter 5.10**, and **Chapter 15.5**.

---

2. The Outlier Data button displays information on each beat and the values that are used for outlier determination. Further details can be found in **Chapter 15.5** and **Figure 42**.

---

3. The 'Cutoff' textbox sets the value of `modz_cutoff` in `Annoparams.m` which determines the modified Z-score cutoff that is used to determine if a beat is an outlier. See **Chapter 15.5** for further details.

---

4. ☑ Auto sets if outliers are automatically removed when **Calculate** has been pressed. If ☐ Auto is unchecked, outliers are marked but not removed.

---

5. The 'X-Corr Thresh' textbox sets the value of `pvcthresh` in `Annoparams.m`. See **Chapter 15.4** for further details.

---

6. The 'RMSE Thresh' textbox sets the value of `rmse_pvcthresh` in `Annoparams.m`. See **Chapter 15.4** for further details.

---

7. Radio buttons that sets the value of `keep_pvc` in `Annoparams.m`. If 'Remove Native' is checked `keep_pvc = 1`, and if 'Remove PVCs' is checked `keep_pvc = 0`. See **Chapter 15.4** for further details.

---

8. The PVC Data button displays a figure that summarizes the data that is used for PVC detection. Further details can be found in **Chapter 15.4** and **Figure 41**.

---

9. the Remove PVCs button removes all beats that have been flagged as PVCs and then recalculates the median beat and associated measurements. Further details can be see in quick start Example 2 (**Chapter 5.2**), **Chapter 5.9**, and **Chapter 15.4**.

---

10. ☑ Auto sets if PVCs are automatically removed when **Calculate** has been pressed. If ☐ Auto is unchecked, PVCs are marked but not removed.

12.11 GUI Section 10 – GUI Toolbar for Interacting with Figures

Section 10:

GUI Toolbar for interacting with figures. The GUI toolbar can be used to navigate within various figures that are displayed in the GUI. Click on an icon in the toolbar and then use in the selected GUI figure.

-  : Pan figure
-  : Rotate figure
-  : Select point in figure
-  : Zoom in on figure
-  : Zoom out of figure
-  : Print figure

13 Batch Processing ECGs with `braveheart_batch.m`

BRAVEHEART allows rapid batch processing of a directory of ECGs via `braveheart_batch.m`. This can be called directly without using the GUI. In batch processing mode there is less graphical output compared to the BRAVEHEART GUI, but when batch processing a large number of ECGs, the batch processing mode of BRAVEHEART is more efficient than manually processing multiple ECGs sequentially in the GUI.

Batch processing of ECGs can be launched from within the GUI (see **Chapter 14**) or via the `braveheart_batch()` function directly from the MATLAB command line. The command line mode is useful for large and complex datasets that may require running multiple batches with different parameters. `braveheart_batch.m` output includes a single `.csv` or `.xlsx` file with ECG/VCG measurements for all batched ECGs, and optional summary figures (See **Figure 50** and **Figure 51**), `.anno` annotation files (see **Chapter 17**), and/or ECG/VCG signals and data in MATLAB `.mat` format (See **Chapter 13.4**).

13.1 Batch Processing Parameters

The parameters that govern how `BRAVEHEART_batch.m` functions can be adjusted at the beginning of the file:

```
% Parameters controlling batch behavior -> Edit as needed

folder = '';                % ECG Folder; set as '' to use system dialog box
format = 'muse_xml';        % ECG format string
output_ext = '.csv';        % Output file extension; '.csv' or '.xlsx'
output_note = 'batch';      % Note added to output

parallel_proc = 0;          % Use parallel processing; 1 = yes, 0 = no
progressbar = 1;            % Display progress bar; 1 = yes, 0 = no

save_figures = 0;           % Save figure of processing/annotation; 1 = yes, 0 = no
save_data = 0;              % Save .mat file of signals/annotations; 1 = yes, 0 = no
save_annotations = 0;       % Save beat annotation file (.anno); 1 = yes, 0 = no

vcg_calc_flag = 1;          % Process VCG_Calc module; 1 = yes, 0 = no
lead_morph_flag = 1;        % Process Lead_Morphology module; 1 = yes, 0 = no
vcg_morph_flag = 1;         % Process VCG_Morphology module; 1 = yes, 0 = no
```

The parameters that can be set to control batch ECG processing include:

`parameter` | *possible values*

parameter description

`folder` | `string`

Sets the path of the folder of ECGs to batch process. If this value is set empty as `''`, the program will bring up a dialog box to allow the user to choose the folder. This parameter is only used when running the command line version of BRAVEHEART from the source code, and is primarily designed for cases when passing in these parameters into the function as shown in **Chapter 13.2**. All other ways of running batch processing (via compiled command line version or via GUI) require the folder input to be via dialog box.

`format` | `string`

String which describes the ECG format to be used. See **Chapter 23** for details of included formats. Formats that are included by default include:

- `'muse_xml'` - GE MUSE .xml
- `'bidmc_format'` - Space delimited ASCII .txt (legacy BIDMC format)
- `'prucka_format'` - GE Prucka recording system ASCII .txt
- `'ISHNE'` - International Society for Holter & Noninvasive Electrocardiology .ecg
- `'DICOM'` - DICOM .dcm
- `'hl7_xml'` - HL7 .xml
- `'philips_xml'` - Philips .xml
- `'mrq_ascii'` - GE Marquee ASCII .mrq
- `'unformatted'` - Unformatted column data in units of mV .txt
- `'cardiosoft_xml'` - Cardiosoft .xml
- `'schiller_xml'` - Schiller .xml
- `'scp_ecg'` - SCP-ECG .scp
- `'edf'` - European Data Format .edf
- `'claris'` - Abbott Workmate Claris ASCII .txt
- `'physionet_csv'` - Physionet .csv
- `'physionet_dat'` - Physionet .dat (and .hea)
- `'megacare_xml'` - Megacare .xml
- `'edan_dat'` - Edan .dat
- `'braveheart_mat'` - BRAVEHEART formatted .mat

`output_ext` | `'.csv'` or `'.xlsx'`

Sets the format of the output file that contains all measurements after batch processing is complete. If set to `'.csv'` the program will generate a comma separated values file, and if set to `'.xlsx'` it will generate a Microsoft Excel spreadsheet file. The information contained in the output file is the same with either output format.

`output_note` | `string`

Sets a string that is included in the measurement output file. This can be used to indicate something about the dataset that is being processed.

`parallel_proc` | `0` or `1`

Sets if MATLAB's parallel computing toolbox is used to process ECGs in parallel. If `parallel_proc = 1` and the user does not have MATLAB's parallel computing toolbox installed, the ECGs will process sequentially without throwing an error. When parallel computing is used, the program uses the maximum number of workers that are available.

`progressbar` | `0` or `1`

Sets if a progress bar pop-up is generated to show the progress of ECG processing.

`save_figures` | `0` or `1`

Sets if output includes a summary figure for each processed ECG. If the number of ECGs to process is large and if figures are not needed, setting `save_figures = 0` can significantly reduce the time required for ECG processing. Examples of summary figures can be seen in **Figure 50** and **Figure 51**.

`save_data` | `0` or `1`

Sets if output includes the raw and filtered signals (individual beats and median beats), annotations, and calculated parameters saved as a `.mat` file. If this data is not needed, setting `save_data = 0` can speed up ECG processing.

`save_annotations` | `0` or `1`

Sets if output includes annotation files for each processed ECG. Annotation files (extension `.anno`) contain the annotation parameters used to process the ECG (from `Annoparams.m` - see **Chapter 6**) and individual beat fiducial point annotations for each ECG. If an `.anno` file is placed in the same directory as an ECG with the same name (eg `ecg_name.xml` and `ecg_name.anno`), the BRAVEHEART command line program overrides the nominal annotation parameters and beat annotations, loading these from the `.anno` file instead.

`.anno` files therefore allow fine control over individual ECG processing via `braveheart_batch.m` without needing to use the GUI. For further details on `.anno` files see **Chapter 17**. `.anno` files behave slightly differently when used with the GUI and more information on use of `.anno` files and the GUI can be found in **Chapter 17**.

`vcg_calc_flag` | `0` or `1`

Sets if data from the VCG Calculation results class `VCG_Calc.m`, which primarily includes vectorcardiographic measurements, is included in the measurement output file. Further details of the data included in the VCG Calculation results class can be found in **Chapter 24**.

`lead_morph_flag` | `0` or `1`

Sets if data from the Lead Morphology results class `Lead_Morphology.m`, which primarily includes data on R, S, and T wave measurements, is included in the measurement output file. Further details of the data included in the Lead Morphology results class can be found in **Chapter 24**.

`vcg_morph_flag` | `0` or `1`

Sets if data from the VCG Morphology results class `VCG_Morphology.m`, which primarily includes vectorcardiographic morphology measurements, is included in the measurement output file. Further details of the data included in the VCG Morphology results class can be found in **Chapter 24**.

13.2 Batch Processing via MATLAB Command Line

Once the parameters noted above in **Chapter 13.1** have been set at the beginning of `braveheart_batch.m`, the program is ready to run. In MATLAB, type `braveheart_batch()` in the MATLAB command window and press Enter. If `folder = ''`, a dialog box will appear to choose the directory which contains the ECGs to be processed, otherwise the specified folder is used. The script will then loop through all of the ECGs with format `format` and the appropriate file extension in the specified directory until complete. Once processing is complete a message will display in the command window noting the number of ECGs processed and if there were any errors during ECG processing. The output will include the folders and files as noted in **Chapter 13.4**.

Alternatively, the parameters can be passed directly into `BRAVEHEART_batch.m` as:
`BRAVEHEART_batch(folder, source_str, output_ext, output_note, parallel_proc, ... progressbar, save_figures, save_data, save_annotations, ...`

`vcg_calc_flag, lead_morph_flag, vcg_morph_flag, aps`) where the first 12 inputs are as noted above, and `aps` passes in a set of annotation parameters. If `aps` is not passed in (only 12 inputs are specified), the program uses the default values contained within `Annoparams.m`. Passing in specific annotation parameters allows the user to run multiple batches with different annotation parameters wihtout needing to manually reset each time.

To construct an `Annoparams` class simply call the `Annoparams` constructor without any inputs. Next, to change a specific annotation parameter, assign it a new value as follows:

```
aps1 = Annoparams();
aps1.lowpass = 0;
aps1.pkthresh = 90;
aps1.STend = 55;
...
```

13.3 Batch Processing Using the Command Line Stand-Alone Executable

The parameters that need to be set when running the executable version of `braveheart_batch.m` are the same as when running it via source code (see **Chapter 13.1** for detailed descriptions of the variables that must be set). The only difference is that because `braveheart_batch.m` is not directly editable when the program is run via executable, the parameters are edited in an external file, `batch_settings.csv`, where the first column is the parameter name, and the second column is the parameter value (with columns separated by commas).

Open `batch_settings.csv`:

```
format,             muse_xml
output_ext,           .csv
output_note,         batch
parallel_proc,         1
progressbar,           1
save_figures,          1
save_data,             1
save_annotations,      1
vcg_calc_flag,         1
lead_morph_flag,       1
vcg_morph_flag,        1
```

and simply edit the values in column 2 as noted in **Chapter 13.1**. Unlike when setting string variables in MATLAB, the string values for `format`, `output_ext`, and `output_note` should

not be enclosed in quotes. The exact order of variables in `batch_settings.csv` is not important, but all 11 variables must be included with appropriate values to avoid any errors. As noted in **Chapter 13.1**, the parameter `folder` is only used when running BRAVEHEART via source code in MATLAB, and it is therefore not set in `batch_settings.csv`. When running the command line version of BRAVEHEART via executable, the ECG folder can only be chosen via dialogue box after starting the program, and the annotation parameters can only be set via editing `Annoparams.csv` or or using the GUI.

Once the parameters noted above in **Chapter 13.1** have been set in `batch_settings.csv` and annotation parameters have been set in `Annoparams.csv` (see **Chapter 6**) make sure these files are saved. The program is now ready to run. Run the executable file. Choose the directory which contains the ECGs to be processed. The script will then loop through all of the ECGs with format `format` in the chosen directory until complete. Once processing is complete text will display how many ECGs were processed, the number of ECGs that were flagged for quality (see **Chapter 16**), and the number that caused errors. The output will include the folders and files as noted in **Chapter 13.4**.

13.4 `braveheart_batch.m` Output

Once batch processing is complete, the output will include:

- `processed_ecgs_<timestamp>.csv` (or `.xslx`)

  Contains electrocardiographic/vectorcardiographic measurements for every ECG that was processed from the directory (1 row per ECG). If specific results classes are disabled (e.g. `vcg_calc_flag = 0`), those values are replaced with `NaN`.

- Folder `figures_<timestamp>/`

  Location where summary figures (see **Figure 50**) are saved if `save_figures = 1`.

  - Summary figure files are named `<ecg_name>.png`.

- Folder `data_<timestamp>/`

  Location where signal and annotation data are stored if `save_data = 1`.

  - Data files are named `<ecg_name>.mat`. Details on the contents of this data structure can be found in **Chapter 25**.

- Folder `annotations_<timestamp>/`

  Location where annotation files are stored if `save_annotations = 1`.

  - Annotation files are named `<ecg_name>.anno`.

`check_ecg_list_<timestamp>.csv`

Lists ECGs that tripped the quality detector and the specific reason for being flagged (see **Chapter 16**) .

Folder `figures_<timestamp>_check_ecg_list/`

Folder where summary figures are copied for ECGs whose annotations tripped the quality detector (see **Chapter 16**).

`errors_<timestamp>.csv`

Lists ECGs that threw an exception during processing and information on the specific exception.

14 Batch Processing ECGs via GUI (`braveheart_gui.m`)

BRAVEHEART can process a batch of ECGs within the same directory from the command line or GUI versions of BRAVEHEART. For information on running BRAVEHEART on a batch of ECGs from the command line version of BRAVEHEART see **Chapter 13**. The outputs and functions of batch processing are identical when batch processing ECGs from the command line or GUI; the GUI is just a wrapper for passing parameters into the command line batch processing program `braveheart_batch.m`.

When using the GUI for batch processing, the settings within the GUI (filtering, annotation parameters, etc.) are used as the annotation parameters (see **Chapter 12** and **Chapter 6.2**) instead of pulling data from `Annoparams.m` (or `Annoparams.csv` if running via executable).

When using the GUI, batch ECG processing can be controlled from GUI Sections 1, 6.2, and 6.6. Parameters in other sections are used to set annotation parameters as noted in prior sections, which is similar to setting values within `Annoparams.m` when using the command line version of BRAVEHEART for batch processing.

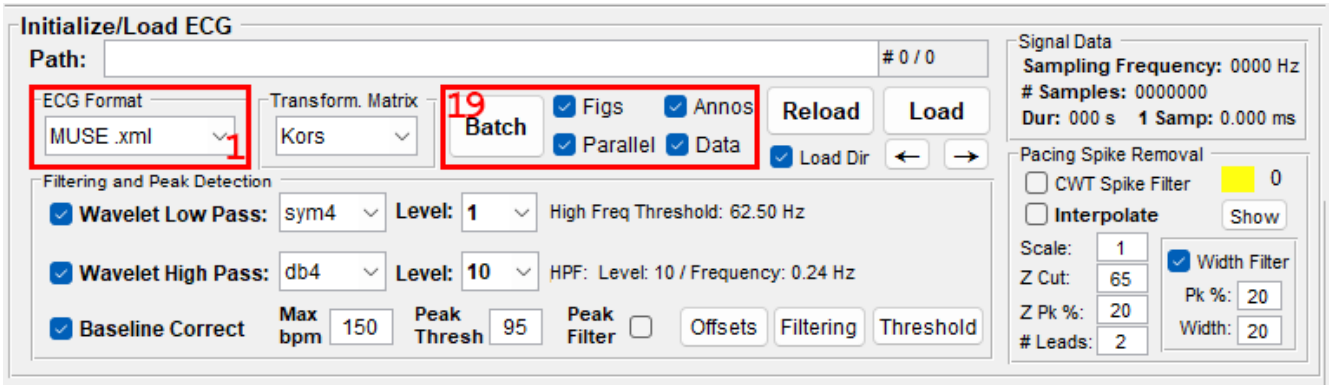


Figure 21: GUI items used to control batch processing in GUI section 1.

`format` is set using the 'ECG Format' dropbox box in GUI item **1.1**.

Instead of setting parameters to control batch processing in `braveheart_batch.m`, in the GUI these are set using checkboxes in GUI item **1.19**:

- ☑ Figs – sets value of `save_figures` as `1` (checked) or `0` (unchecked).
- ☑ Parallel – sets value of `parallel_proc` as `1` (checked) or `0` (unchecked).
- ☑ Annos – sets value of `save_annotations` as `1` (checked) or `0` (unchecked).
- ☑ Data – sets value of `save_data` as `1` (checked) or `0` (unchecked).

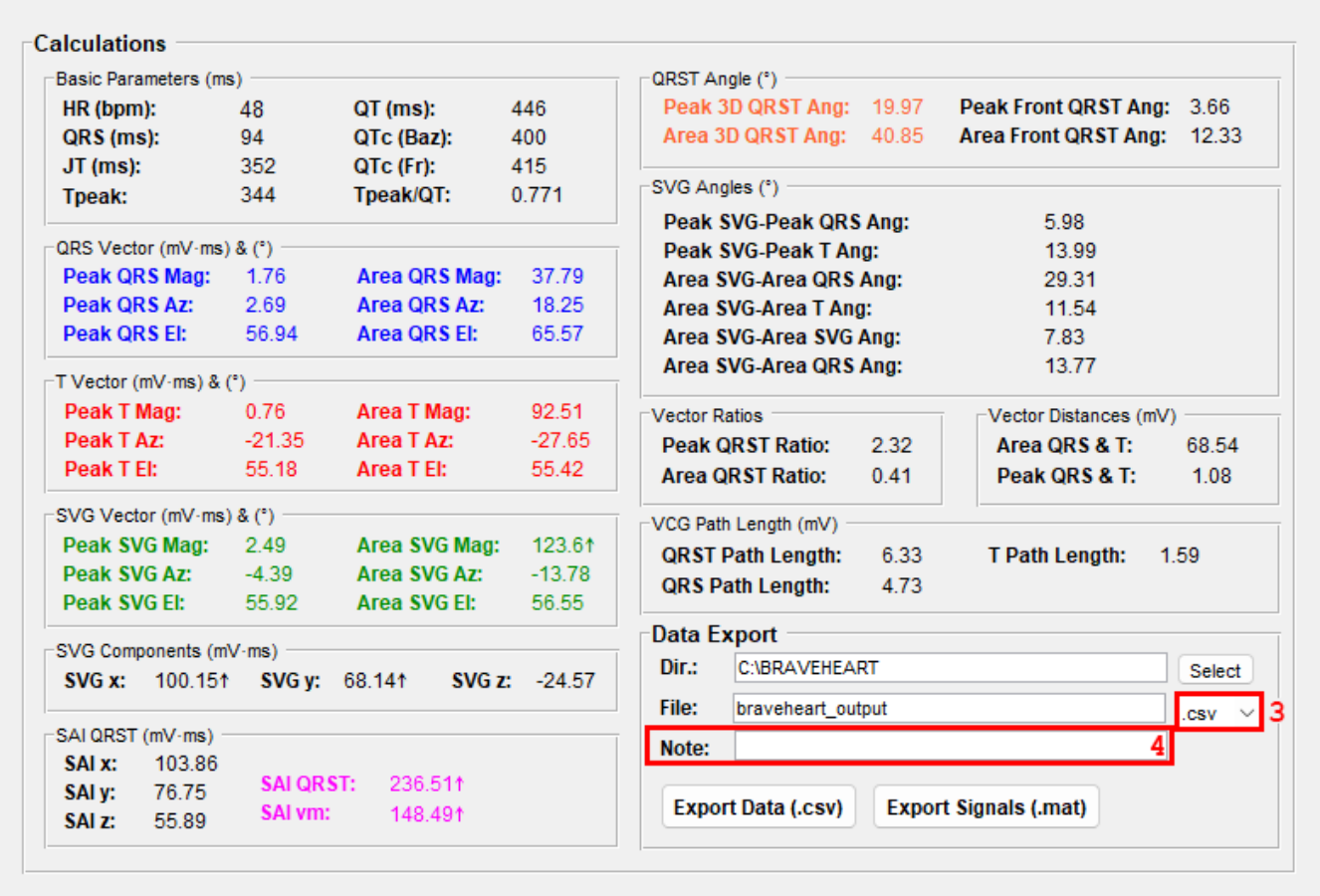


Figure 22: GUI items used to control batch processing in GUI Section 6.2.

- `output_ext` is set using the VCG Calculations section in dropdown GUI item 6.2.3.
- `output_note` is set using the VCG Calculations section in the 'Note:' textbox GUI item 6.2.4.

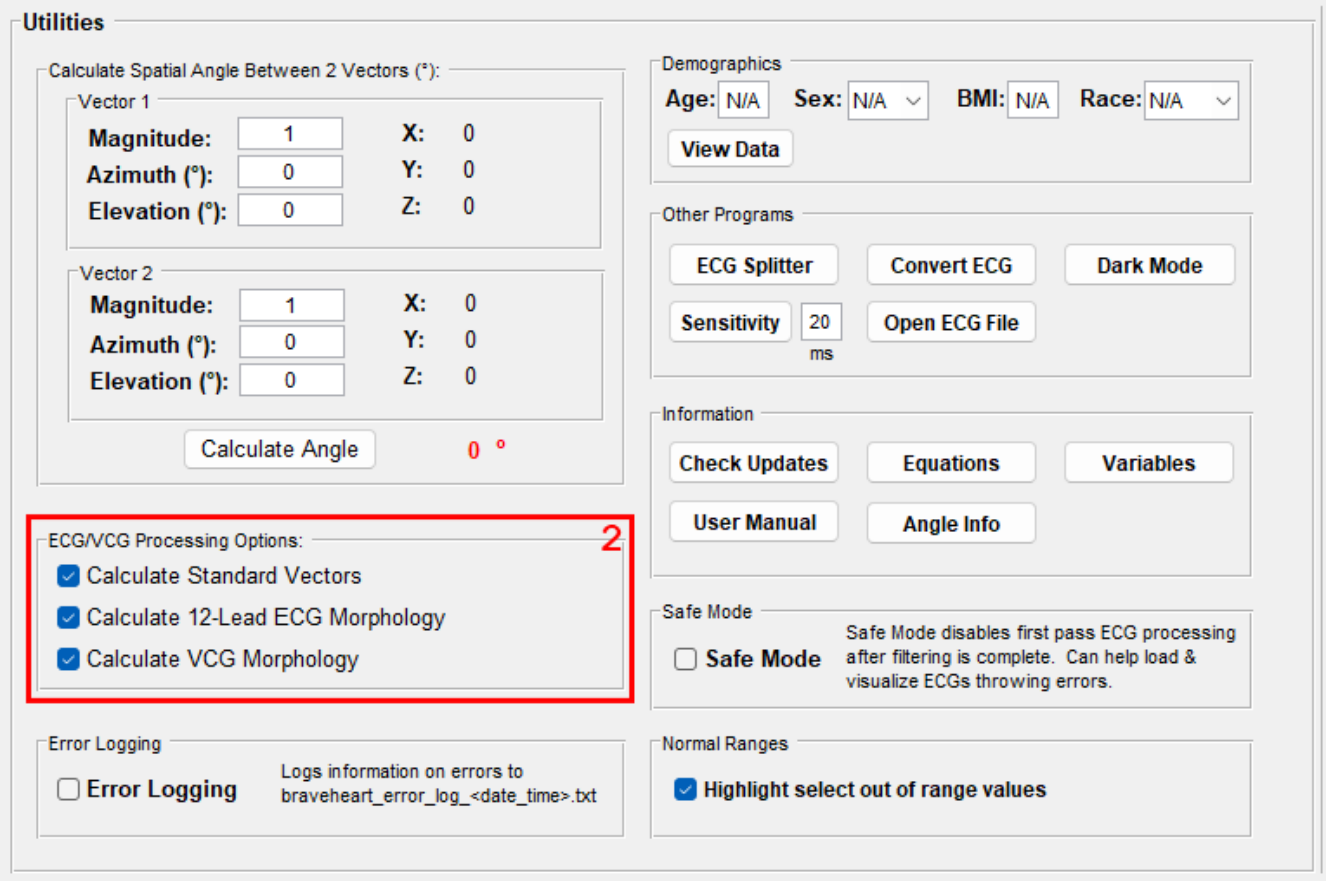


Figure 23: GUI items used to control batch processing in GUI Section 6.6.

Control over what data is processed is set by checkboxes in the Utilities section and GUI item 6.6.2:

- ☑ Calculate Standard Vectors – sets value of `vcg_calc_flag` as `1` (checked) or `0` (unchecked).
- ☑ Calculate 12-Lead ECG Morphology – sets value of `lead_morph_flag` as `1` (checked) or `0` (unchecked).
- ☑ Calculate VCG Morphology – sets value of `vcg_morph_flag` as `1` (checked) or `0` (unchecked).

There is no way to disable the progress bar via the GUI.

Once all batch parameters and annotation parameters have been set in the GUI, click on the Batch button (GUI item 1.19). A dialog box will appear; select the directory that contains the ECGs and click 'Open'. Batch will begin. A progress bar displays the progress of batch ECG processing. Once finished, a pop-up will display how many ECGs were processed, the number of ECGs that were flagged for quality (see **Chapter 16**), and the number that caused errors. Output directory structure and file naming conventions are the same as noted in **Chapter 13.4**.

15 BRAVEHEART ECG/VCG Processing

15.1 R Peak Detection

QRS Complexes (R peaks) are nominally detected in the vector magnitude (VM) lead by finding peaks in the top n% of values (where $n =$ `pkthresh` in `Annoparams.m`) subject to a minimum R-R interval constraint (`maxBPM` in `Annoparams.m`) which helps prevent incorrect detections of fractionated R peaks within the same QRS or large amplitude T waves. The VM lead is used for R peak detection because it is always positive, and the QRS complex peak amplitude is usually significantly larger than the T wave peak amplitude. In general, the nominal value for `peakthresh = 95` works well for most applications, but may have to be slightly adjusted in ECGs that have significant artifact (see **Chapter 31**), ECGs with ectopic/paced beats with a large range of QRS amplitude in different beats, or ECGs with very large amplitude T waves. The value of `pkthresh` may also have to be adjusted based on ECG sampling rate or length, as these parameters affect the number of samples in the ECG and therefore also affect how the sample values are binned into percentiles.

Figure 24 below illustrates how the parameters `peakthresh` and `maxBPM` can be adjusted for R peak detection. Issues with R peak detection (usually due to large artifact) are a common cause of ECGs failing to load properly (see **Chapter 31**).

As of version 1.3.0, there is now an option to filter the VM lead, primarily to remove T waves, buy specifying `pkfilter = 1` in `Annoparams.m`, or by selecting ☑ peak Filter in GUI item 1.8. Peak filtering is accomplished by using wavelet decomposition of the VM signal, and then discarding the approximation coefficents (always very low frequency) and details coefficients that contain frequencies outside of 10-40 Hz (the frequency content of the QRS complex which is not usually present in P or T waves). The filtered signal is then reconstructed by using only the details coefficients that correspond to 10-40 Hz. The signal is then squared and peaks are detected above a threshold which is set similarly to the existing R peak detection algorithm.

The peak filtering option is off by default since in general, the T wave amplitude is significantly less than the QRS amplitude in the VM lead. In certain situations with very peaked T waves or low frequency noise, enabling peak filtering may be useful. The value of `pkthresh` may have to be adjusted when using peak filtering because removal of the T waves often results in a signal with sudden large amplitude peaks timing with QRS complexes and very small amplitude signals between R peaks. This affect can be seen in **Figure 25** below.

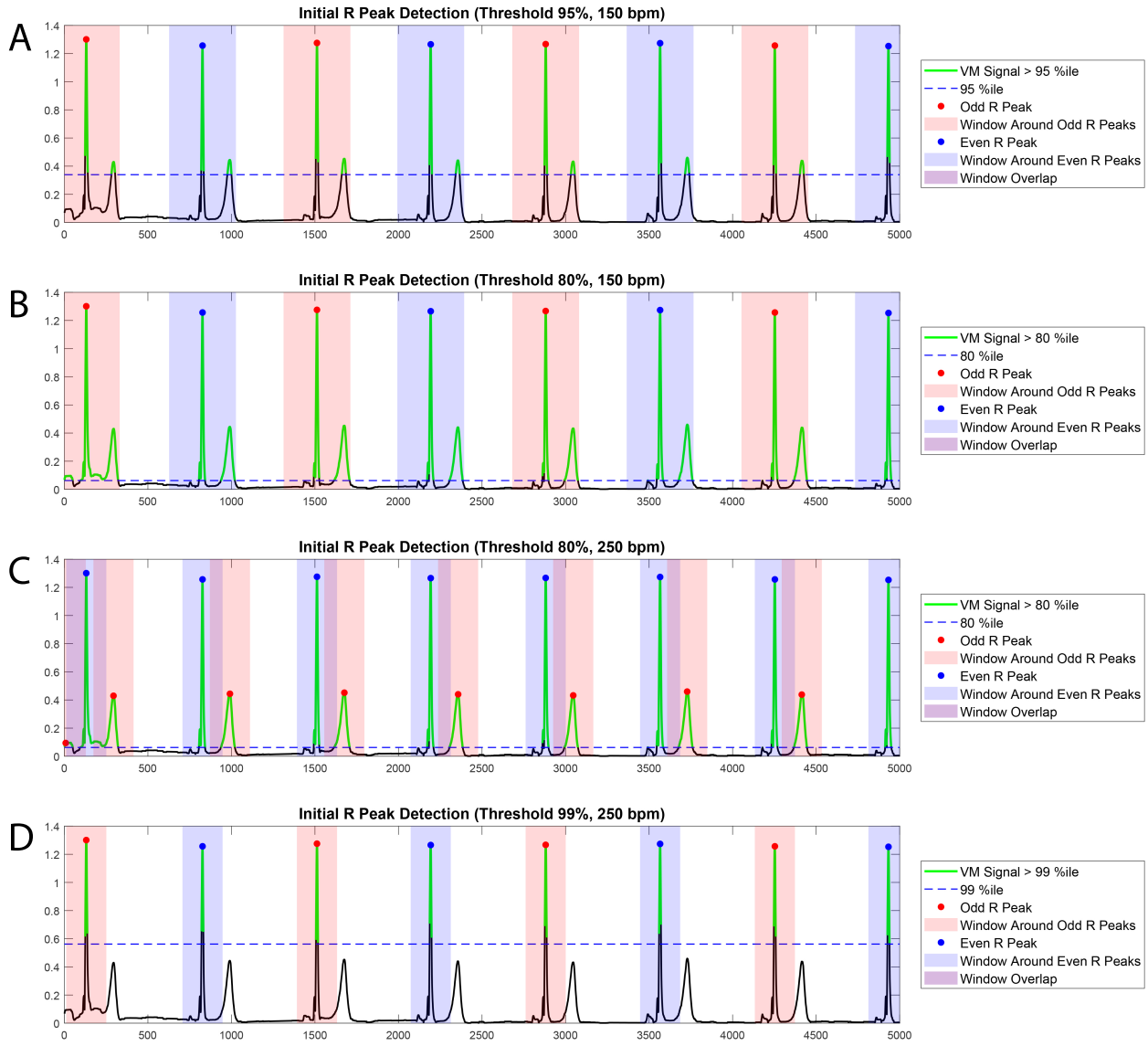


Figure 24: Examples of how peak thresholding (`pkthresh`) works in the context of heart rate constraints (`maxBPM`). The same results can be obtained with different combinations of `pkthresh` and `maxBPM` . In all of these examples, `pkfilter = 0` and peak filtering is disabled. **A:** Nominal settings with `pkthresh = 95` and `maxBPM = 150` . The R peaks are appropriately identified (red and blue circles), but even though the T wave peak is also above the 95th percentile (green signal above the blue dashed line), it is not identified as a R peak because it falls within the blanking period (red and blue shaded areas) defined by `maxBPM` . In this case, `maxBPM = 150` corresponds to an interval of 400 ms or 200 samples at 500 Hz on each side of the detected R wave. **B:** The same ECG with `pkthresh = 80` . The R peak detection does not change because the majority of the QRST complex that is above the 80th percentile (green signal above dashed blue line) still falls with the appropriate blanking period. **C:** In this example `pkthresh = 80` but `maxBPM = 250` . In this case, because the blanking windows are significantly shorter (120 samples on each side of each detected peak), the peak of the T wave which is above the 80th percentile (green signal above dashed blue line) is now detected as a R peak (red dots). **D:** Using `maxBPM = 250` , R peaks are now appropriately identified without detecting T wave peaks by increasing the threshold to `pkthresh = 99` . Although the blanking window is short, the T wave peak is now below the 99th percentile (green signal above blue dashed line), so even though the T wave peak falls outside the blanking window, it is no longer detected as an R wave peak because its amplitude is below the 99th percentile (dashed line).

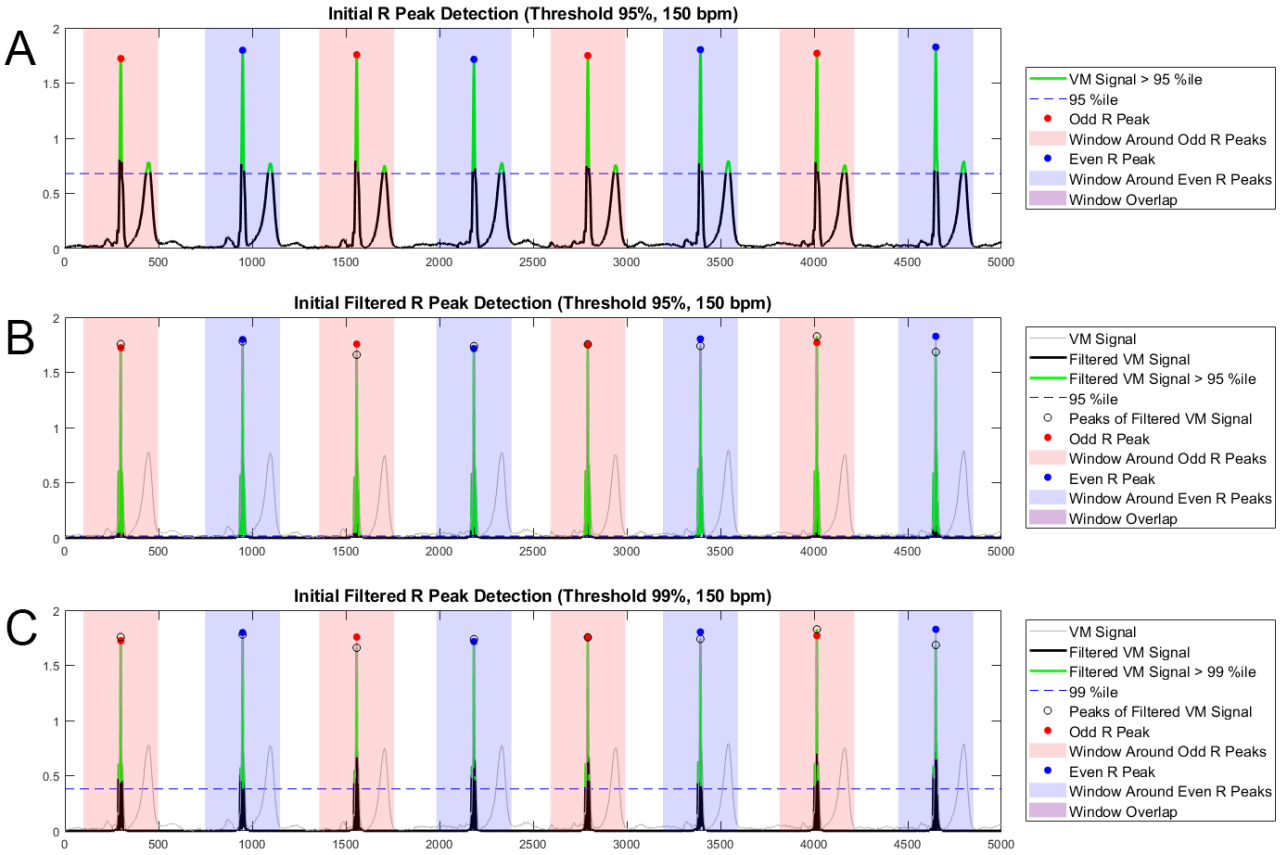


Figure 25: Examples of how peak thresholding works when `pkfilter = 1`. **A:** Nominal settings with `pkthresh = 95`, `maxBPM = 150`, and `pkfilter = 0` (peak filtering off). As seen in panel **A** of **Figure 24**, the R peaks are appropriately identified (red and blue circles), even though the T wave peak is also above the 95th percentile (green signal above the blue dashed line), it is not identified as a R peak because it falls within the blanking period (red and blue shaded areas) defined by `maxBPM`. **B:** The same ECG is processed with the same values of `pkthresh` and `maxBPM`, but peak filtering is now enabled (`pkfilter = 1`). The unfiltered ECG is shown as a thin grey line, and the filtered signal is shown with a thick black line. Note how since the T wave is removed, the signal between T peaks is approximately zero. The 95th percentile for the filtered signal is therefore also very close to zero, and almost the entire R peaks are green, indicating they are above the 95th percentile. Although this did not cause any issues with identifying R peaks for this specific ECG, having the value of `pkthresh` so close to zero can result in false R peak detections due to low amplitude noise. **C:** The same ECG is processed with peak filtering on and a with `pkthresh = 99`. Now the 99th percentile cut off value is further away from zero and there is less chance for false R peak detections. Note that in **B** and **C**, the empty black circles indicate the peaks localized in the filtered signal. When peak filtering is enabled, the R peak finding algorithm uses these values to find the peaks in the original VM signal. Peaks in the original VM signal are always reported as the R peaks regardless of if filtering is enabled or disabled.

15.2 Pacing Spike Removal/Interpolation

Artifact from cardiac pacing may sometimes cause significant issues with ECG processing and annotation, especially if pacing spikes are very large or very wide. BRAVEHEART has 2 methods for dealing with pacing artifacts if needed.

Prior to version 1.5.0, the only way to deal with pacing spikes was to ignore them when detecting

R peaks (see **Chapter 15.1**) using median filtering to ignore the narrow peaks caused by pacing artifacts. As of version 1.5.0, there is a more robust method (using the continuous wavelet transform [CWT]) of dealing with pacing spikes to prevent them from being detected as R peaks (and therefore interfering with QRST annotation), and, if needed, to remove the pacing spikes from the ECG signal. We will refer to this new method available in v1.5.0 forward as the "CWT method" and the original method which is available in all versions, as the "median filter method".

IMPORTANT: This CWT method of identifying/ignoring pacing spikes is significantly slower than the median filter method. If you are processing large data sets that will not have any paced ECGs, we recommend that you disable pacemaker spike detection or use the median filter method.

Median Filter Method

The median filter method of pacemaker spike detection is controlled by the following parameters in `Annoparams.m` which are described in detail in **Chapter 6.2**.

- `spike_removal` which enables (`= 1`) or disables (`= 0`) the method.
- `pacer_mf` which sets the width of the median filter in ms.
- `pacer_thresh` which sets the threshold of spike height (in %) for measuring spike width.
- `pacer_spike_width` which sets the maximum width that defines a pacing spike.

The median filter method of pacemaker spike detection is enabled/disabled by setting `spike_removal` = 1 or 0 in `Annoparams.m`, or checking/unchecking the ☑ Width Filter checkbox in GUI section **1.18**. If enabled, during R peak detection, the signal is smoothed with a median filter nominally set to 4 ms. This value should rarely need to be changed, but it can be adjusted by changing the value of `pacer_mf` in `Annoparams.m`. After median filtering, the start and end of each detected VM R peak is found by marching from the peak forwards and backwards and marking where the signal first crosses an adjustable threshold (set by `pacer_thresh` in in `Annoparams.m` which is nominally set at 20% of the peak height). The width of any detected spike is then calculated as the distance between these start and end positions. Any such peak that has width below a user defined threshold (set by `pacer_spike_width` in `Annoparams.m` with default value 20 ms) are marked as pacing spikes, as such narrow spikes would not be found in physiological QRS complexes. If pacing spikes are found they are ignored and the R-peak finding procedure is repeated until no spikes are detected. The remaining VM R peaks are the QRS R peaks and not pacing spikes.

CWT Method

The CWT method of pacing artifact removal was introduced in BRAVEHEART v1.5.0, and because this method was not included in our original methods manuscript, we are including additional technical details on how the algorithm works in addition to how it can be used.

The CWT method detects/removes pacing artifacts on the raw ECG signals **prior to any filtering**. This was done because low-pass filtering attenuates the high frequency content of the ECG signals, which is what we want to assess during pacing artifact detection, and it was difficult to find generalizable cutoffs for the algorithm with signals having a wide range of filtering. For this reason, variations in hardware filtering at the time of ECG acquisition may require adjustments of parameters controlling the CWT method (see below) even if the sampling frequency is the same.

The CWT method is controlled by the following parameters in `Annoparams.m` which are described in detail in **Chapter 6.2**.

- `cwt_spike_removal` which enables (`= 1`) or disables (`= 0`) the CWT method.
- `interpolate` which enables (`= 1`) or disables (`= 0`) spike interpolation.
- `pacer_zcut` which sets the threshold for spike detection on the modified Z score signal.
- `pacer_zpk` which sets the percent of modified Z score peak at which peak onset/offset are defined.
- `pacer_maxscale` which sets the upper limit of the CWT scale used for detection.
- `pacer_spike_num` which sets the minimum number of leads that have to have pacing spikes detected.

The 12 raw ECG leads are mirrored to reduce edge effects and the CWT is calculated using the analytic Morse wavelet. This decomposes the signal into different scales which correspond to various pseudo-frequencies. The lowest scales contain the highest frequency coefficients. The coefficients from the first n scales are then used for analysis, where n is set by the value of `pacer_maxscale` in `Annoparams.m` or the GUI. The absolute value of the coefficients are summed and then the modified Z score of the 1st difference (Z) is calculated.

Peaks in Z with a minimum height of `pacer_zcut` are then located. If no peaks are found, the lead is considered "not paced". If at least 1 peak is found the lead is considered "paced" and additional processing is performed. The start and end of each peak is found by marching from the peak forwards and backwards and marking where the signal first crosses a value defined by `pacer_zpk` % of the peak height. After peak locations, peak starts, and peak ends are located in all 12 leads, the locations of starts/ends are combined to adjust for each lead potentially

having a different pacing spike morphology and width. Once a spike is detected in any lead, it is assumed that a spike exists in the other 11 leads even though it may not have been detected as a Z spike due to low amplitude in some leads. Hyperbolic cosine interpolation is used to remove the ECG signal between each peak start and end. The resulting signals therefore have the detected pacing spikes removed.

Once the spikes are removed, if `interpolate = 0`, the interpolated signal without pacing spikes is only used for R peak detection similar to how the "median filter" method operates. Once R peak detection has been performed, the signal with the pacing spike in pace is used for subsequent calculations. Because the pacing spike remains in the ECG signal, median beat annotation should usually annotate QRS onset after the pacing spike. If `interpolate = 1`, the interpolated signal is used for all subsequent processing/calculation, with the goal of not seeing any pacing spikes in the processed VCG.

The CWT method has numerous advantages over the median filter method for spike removal. The CWT method can detect and remove multiple pacing spikes, as well as pacing spikes that start after the QRS onset and which distort the early part of the QRS complex. The CWT method is also much more robust to dealing with very large pacing artifacts that sometimes fail to be removed with the median filter without adjustment of parameters. The major downside to the CWT method is that the CWT requires significant additional computation time, and ECG processing is therefore slower. Unintended affects of signal interpolation also need to be considered - ideally only a small section of each lead will be interpolated, but depending on the ECG frequency and annotation parameter settings, excess signal may be interpolated which is not ideal. Likewise, if the annotation parameters are set incorrectly (especially the peak width (`pacer_zpk`) the area of interpolation can be too wide or too narrow, introducing additional artifacts.

As of v1.5.0 there are some additional variables related to pacing spikes that are saved during ECG processing and which can be accessed outside of BRAVEHEART. See **Chapter 25** for further details.

CWT Method Experiments

We performed some experiments to determine the optimal settings for the CWT method of pacemaker spike removal. It should be noted that because the CWT method operates on raw, unfiltered ECG signals, the exact settings needed and the output of any specific set of settings will be influenced by the ECG sampling frequency and the hardware filtering at the time the ECG is acquired. As noted above, using the raw signals allows more standardized parameters

than if filtered signals were used.

We started out by taking sets of ECGs acquired at 1000 Hz, 500 Hz, and 250 Hz, excluding any ECGs with pacing spikes, and measuring the maximum values of the modified Z score in each lead at different maximum scales (corresponding to different values of `pacer_maxscale`) as shown in the figure below:

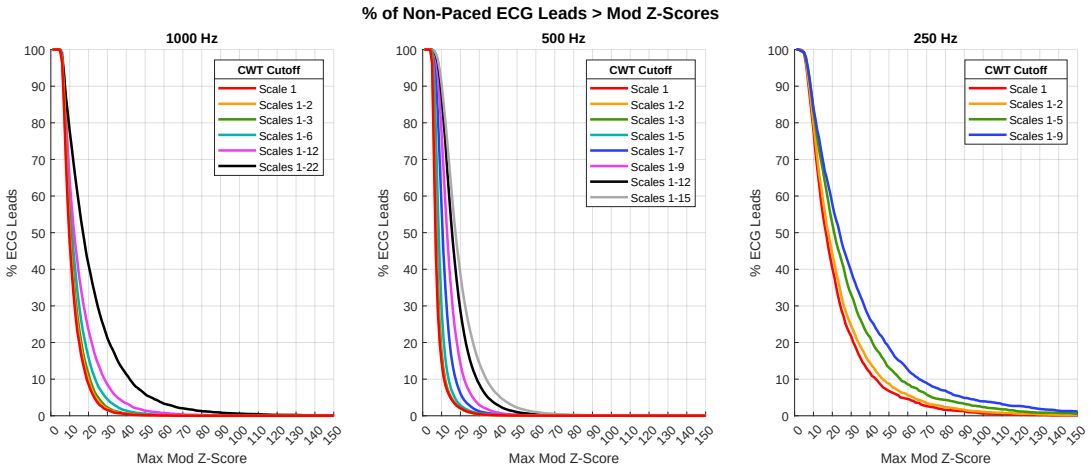


Figure 26: Percent of ECG leads with different maximum modified Z scores for ECGs without any pacing artifacts present sampled at 1000 Hz (N=6,048), 500 Hz (N=20,928), and 250 Hz (N=2,244). The highest frequency components are contained in the lower scales. As can be seen, as the number of scales included is increased and additional lower frequency signal components are included, there is an increase in the overall maximum modified Z score. Given their lower sampling frequency, ECGs sampled at 250 Hz had higher maximum modified Z scores compared to ECGs sampled at higher frequencies. The differences between values at 500 Hz and 1000 Hz is likely due to variations in hardware filtering at the time of ECG acquisition.

As can be seen, at all sampling frequencies, excluding higher scales (lower frequency components) results in lower maximum values of the modified Z score in each lead, and the smallest values are seen when only the lowest scale (scale 1) was used. For sampling frequencies of 500 Hz and 1000 Hz, there was not a significant difference when using scale 1 alone, or using scales 1-2 or 1-3 as can be seen when we zoom in on the above figure:

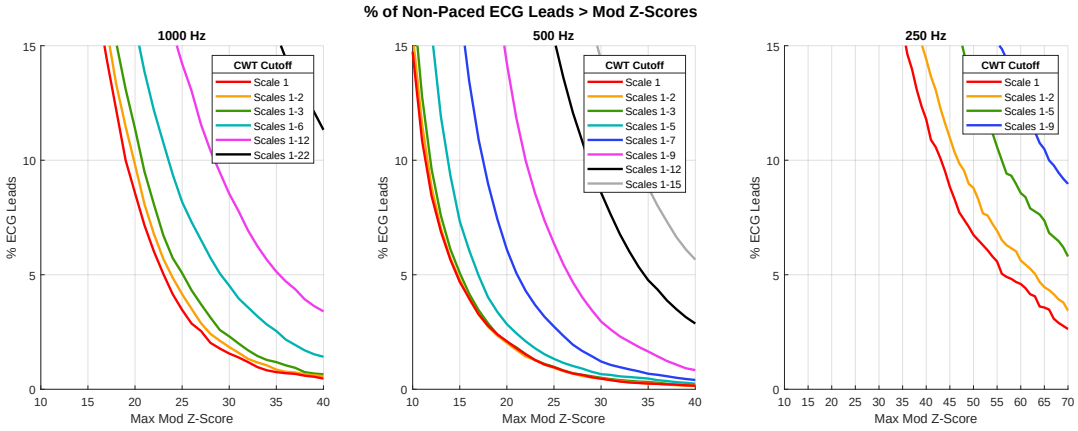


Figure 27: **Figure 26** zoomed in.

It should be noted that ECGs sampled at 250 Hz tended to have significantly higher modified Z scores values compared to ECGs with higher sampling rates. This is expected as there is less distinction between the high frequency components outside the QRS complex frequencies at lower sampling frequencies.

When scale 1 was compared between ECGs sampled at 1000 Hz, 500 Hz, and 250 Hz (see **Figure 28**), it can be seen that setting the minimum peak height in Z to 65 (`pacer_zcut = 65`) results in 99.97% of ECG leads without pacing not having a peak detected in Z. It was not possible to get this level of discrimination with ECGs sampled at 250 Hz, and if ECGs sampled at $<$500 Hz are used, the value of `pacer_zcut` will have to be adjusted to closer to $>$100.

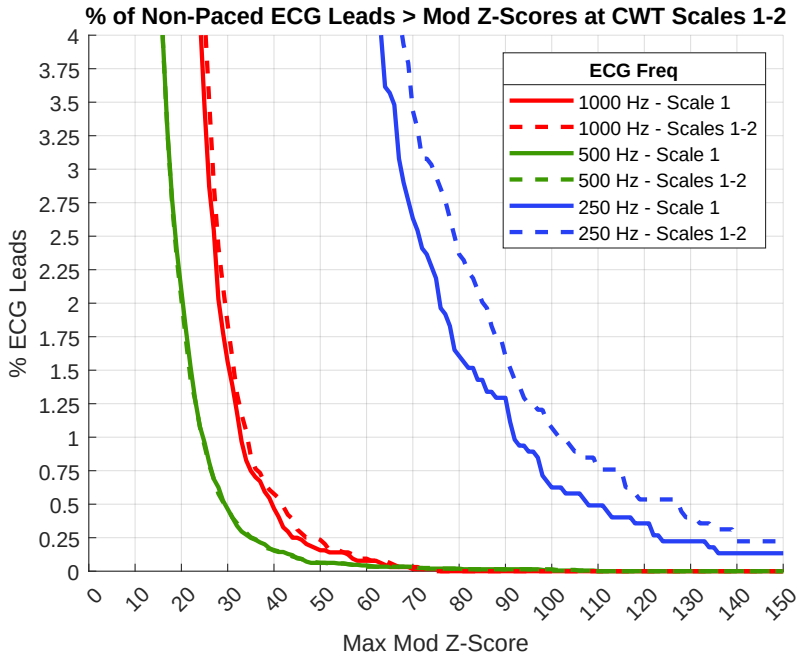


Figure 28: Comparison of using only scale 1 or scales 1-2 for ECG sampled at 1000 Hz, 500 Hz, and 250 Hz. As can be seen, there is minimal difference for ECGs sampled at 500 Hz and 1000 Hz, but a significant difference for ECGs sampled at 250 Hz. This is likely due to the fact that the scale 1 for the ECGs sampled at 250 Hz has a pseudo-frequency band around 100 Hz, so more of the regular QRS complex is retained in the CWT filtered signal.

For a subset of ECGs sampled at 500 Hz, we then looked at the values of Z at pacing spikes of various amplitudes and compared these values to the maximum values of Z in areas without pacing spikes:

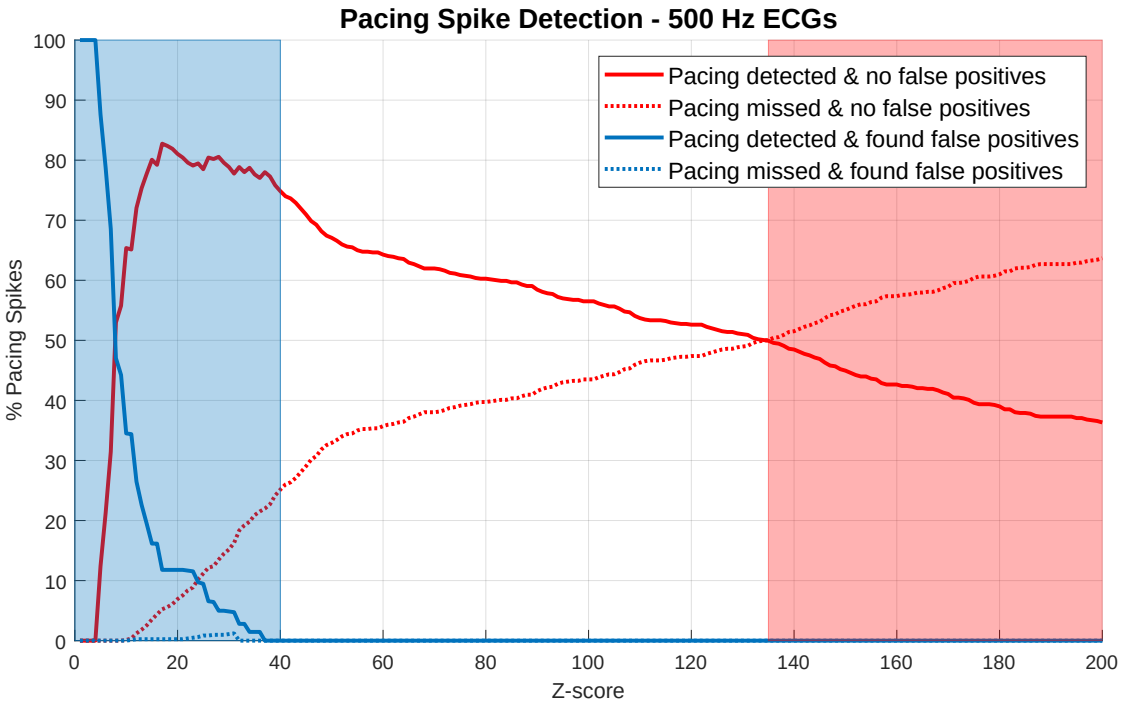


Figure 29: Sample of 3,360 ECG leads from 280 ECGs acquired at 500 Hz with at least 1 pacing spike present, evaluating rates of true and false positive and negative pacing spike detection at various modified Z score cutoffs (`pacer_zcut`). In comparison to the data in **Figure 26**, the maximum value the modified Z score for non-paced areas of the ECG leads (blue lines indicating false positive spike detections) was slightly lower, at approximately 40. This value may be somewhat artificially lower than expected because parts of the rapidly changing QRS complex were partially dominated by the pacing spike and small areas with higher modified Z scores around the pacing spikes were excluded from that analysis for logistical reasons. As can be seen, if `pacer_zcut` is not greater than the maximum value of the non-paced signal, there will be a high rate of false positive spike detection (blue shaded area). Once `pacer_zcut` is above this point, however, there is a range of values (white section) where the rates of true positives and false negatives vary with the cutoff point chosen. The optimal value of `pacer_zcut` is just above the maximum value of the non-paced signals (end of blue shaded area), although this value may be higher depending on the exact ECG signals used (see **Figure 26**). As `pacer_zcut` is further increased there is a decline in the rate of true positives and a increase in the rate of false negatives until a second point (where the red line and red dashed line cross) where the value of `pacer_zcut` is too high and the rates of false negatives is higher than the rate of true positives (red shaded area).

In this dataset, there was a slightly lower value of the maximum Z scores of non-paced signals (closer to 40). It can be seen that once a value of `pacer_zcut` is above the maximum value of the modified Z score for non-paced signals (about 40 in this case – end of the blue shaded area in the figure above), the rate of false positives is approximately 0. As the value of `pacer_zcut` is increased further the rate of missed spikes increases until there is a second point at which the rate of missed spikes is greater than the rate of correctly identified spikes (start of red shaded area in the figure above).

Based on the above data, we decided to set the nominal value of `pacer_zcut = 65`, understanding that thiw will have to be increased significantly for ECGs with sampling

frequencies < 500 Hz. A value for `pacer_zpk = 20` was chosen empirically to balance wanting to avoid under-interpolation or over-interpolation. For **very** large or wide/multi-component pacing spikes, very low values of `pacer_zpk` may be needed to completely remove the large/wide pacing spike. We are working on a more dynamic method for choosing `pacer_zpk`, and may release an improved and more automatic cutpoint in a future version of BRAVEHEART.

Visualization of the CWT Method

When using the GUI, the parameters and results of pacemaker spike detection and removal using the CWT method can be visualized using figures that can be generated with the Show button in GUI section **1.18**.

This will generate a figure that shows the 12 raw ECG lead signals, the modified Z score of the first difference of the ECG lead signal, and the value of `pacer_zcut` with any detected pacing spikes:

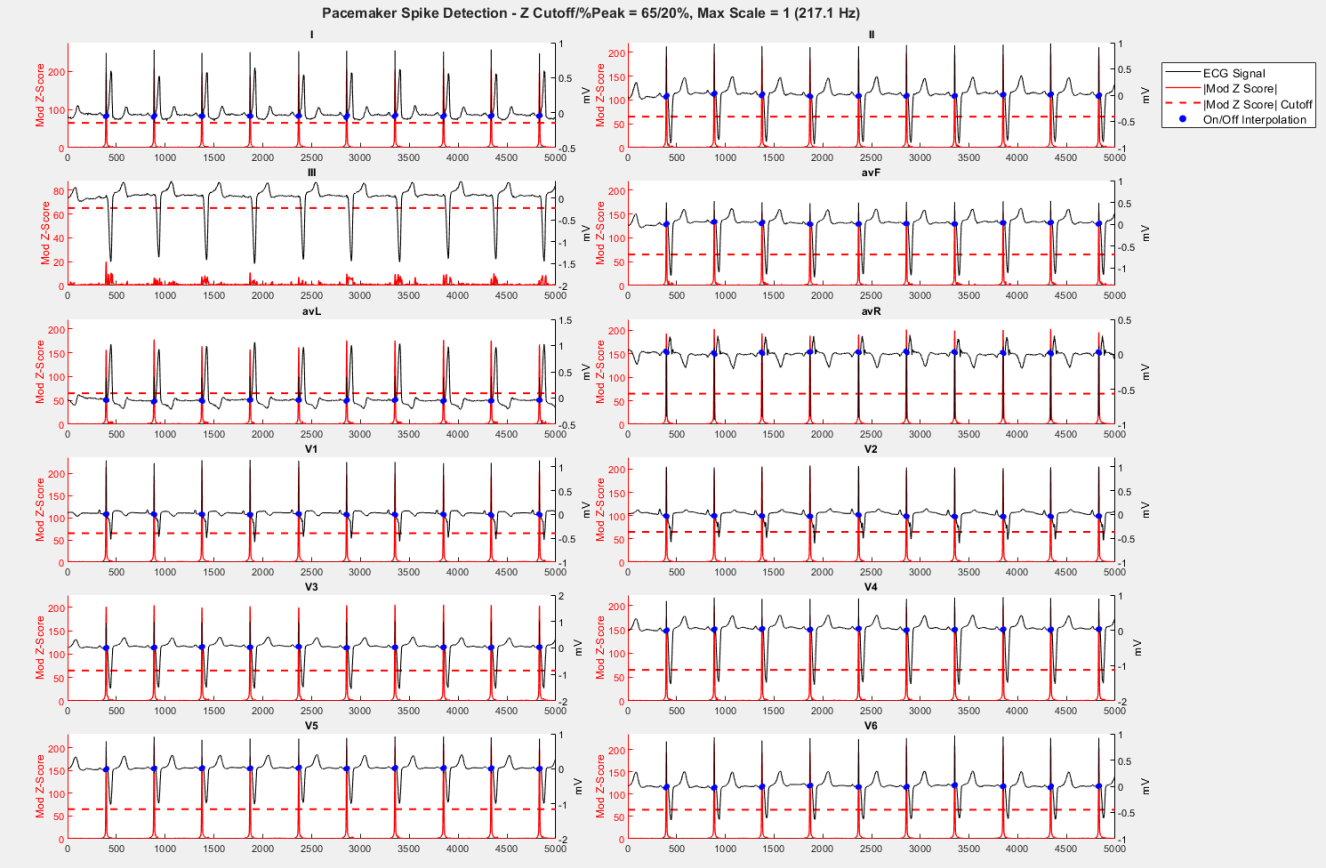


Figure 30

The above Figure **Figure 30** shows data from the ECG `example4.xml`. The raw ECG signals

for each lead are shown in black and the absolute value of modified Z scores for the 1st difference of the ECG leads is shown in red. The value of `pacer_zcut` is shown as the horizontal red, dashed line. The title of the figure shows the numeric values of `pacer_zcut = 65` and `pacer_zpk = 20`, as well as the value of `pacer_maxscale = 1` and the lower limit of the pseudo-frequencies that are included for pacemaker spike detection (217.1 Hz).

Any time the red modified Z score signal crosses the red horizontal dashed line a pacing spike is detected. The onset and offset of the spike, as determined by `pacer_zpk`, are noted by blue dots (will have to zoom in to see):

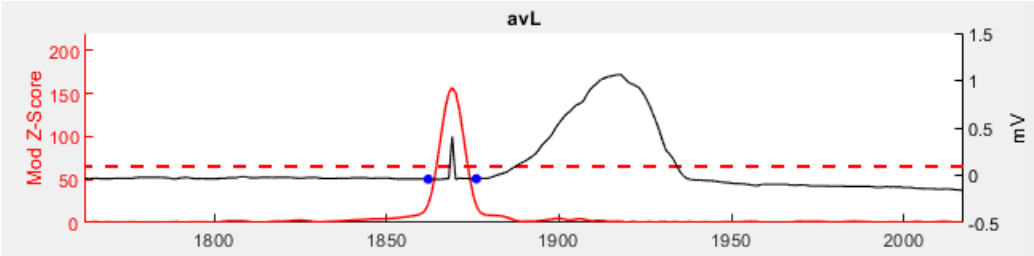


Figure 31

Note that pacing spikes are detected in all leads except lead III. This is why the GUI shows a value of 11 next to the yellow square with the lightning bolt in GUI section **1.18**.

If **☑ Interpolation** is checked or `interpolation = 1` in `Annoparams.m`, a second figure will also appear showing how each ECG lead's pacing spikes were removed with interpolation:

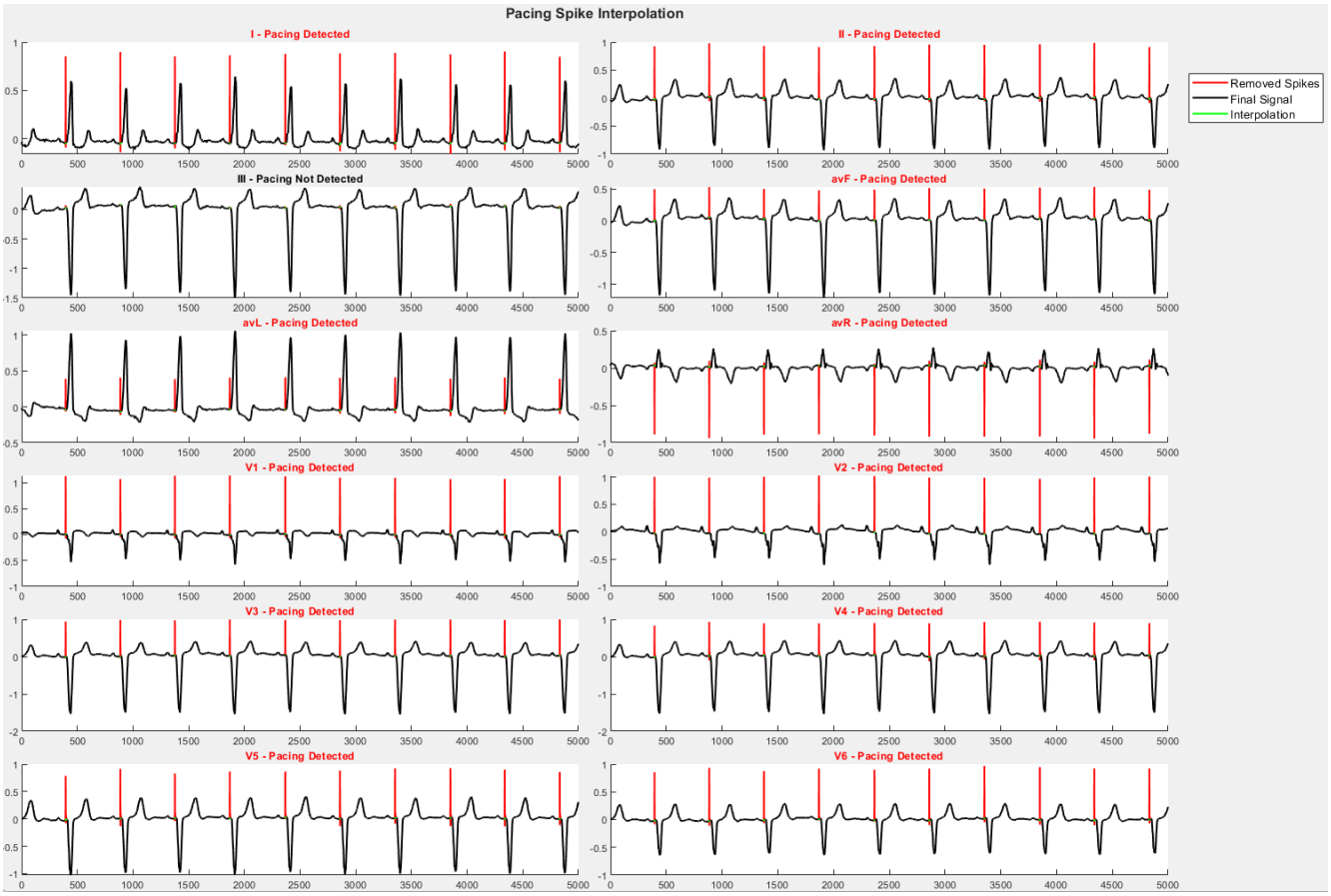


Figure 32

In the above Figure **Figure 32**, the original ECG signal in each lead is shown in black and the pacing spikes that are removed are shown in red. Text above each lead notes if pacing spikes were detected in that specific lead based on the data presented in Figure **Figure 30**. The final signal that is the result of interpolation is shown in green (may have to zoom in to see it):

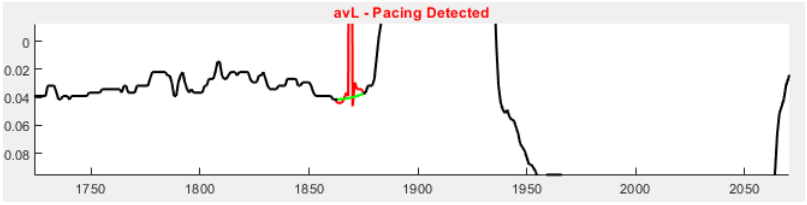


Figure 33

Now consider what would happen to the ECG `example4.xml` if we had set `pacer_zcut = 210` for illustrative purposes.

When we generate the figures we see:

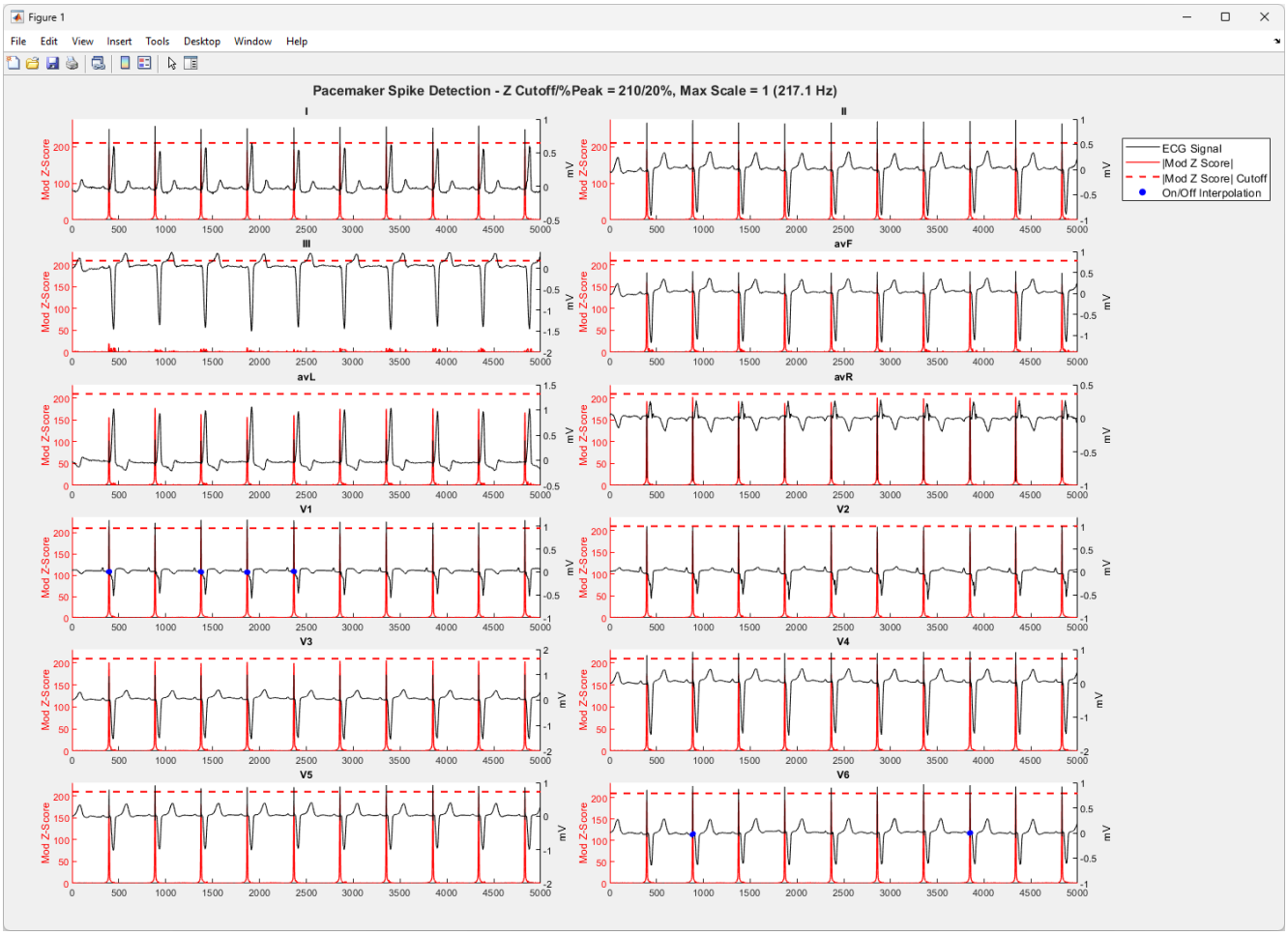


Figure 34

Note that now, with the higher value of `pacer_zcut`, pacing spikes are only detected in lead V1 (beats 1, 3, 4, and 5), and lead V2 (beat 2). Since `pacer_spike_num = 2` and 2 leads had pacing spikes detected, pacing is "detected" and spikes are processed as noted above. If `pacer_spike_num = 3`, there would not have been enough leads with pacing to meet criteria for pacing spike detection, and no further pacing spike processing would have proceeded. As a result, pacing spikes are only appropriately detected and processed to allow correct R peak detection in beats 1-5:

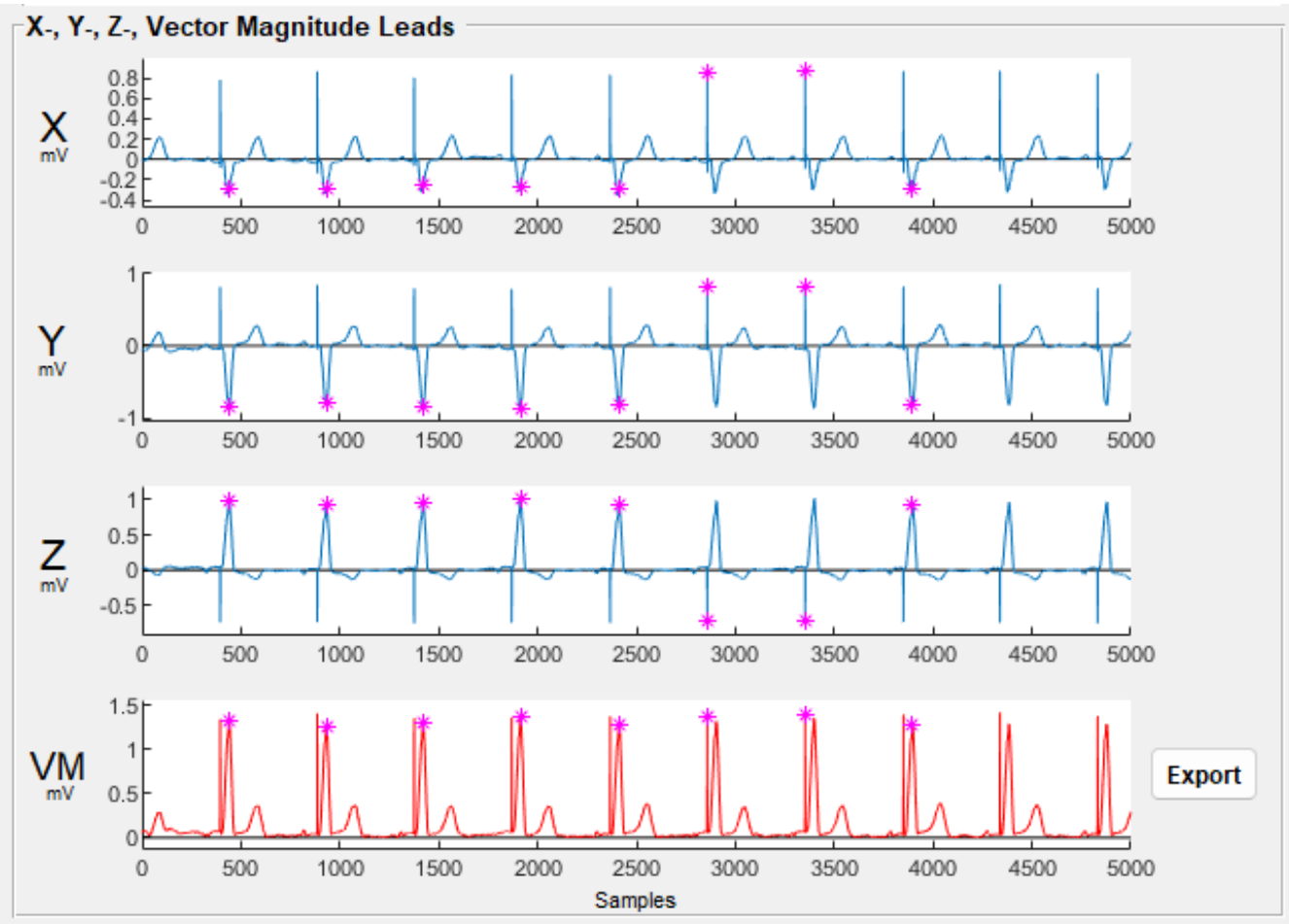


Figure 35

Also note that since we had 2 leads with pacing and that this number met the criteria for the minimum number of leads to have pacing detected (`pacer_spike_num`), beats 1-5 will be declared to having pacing spikes even though beats 1-5 did not have pacing spikes detected in all leads with spikes.

Finally, we will illustrate what happens if we change `pacer_zpk = 20` to `pacer_zpk = 80`:

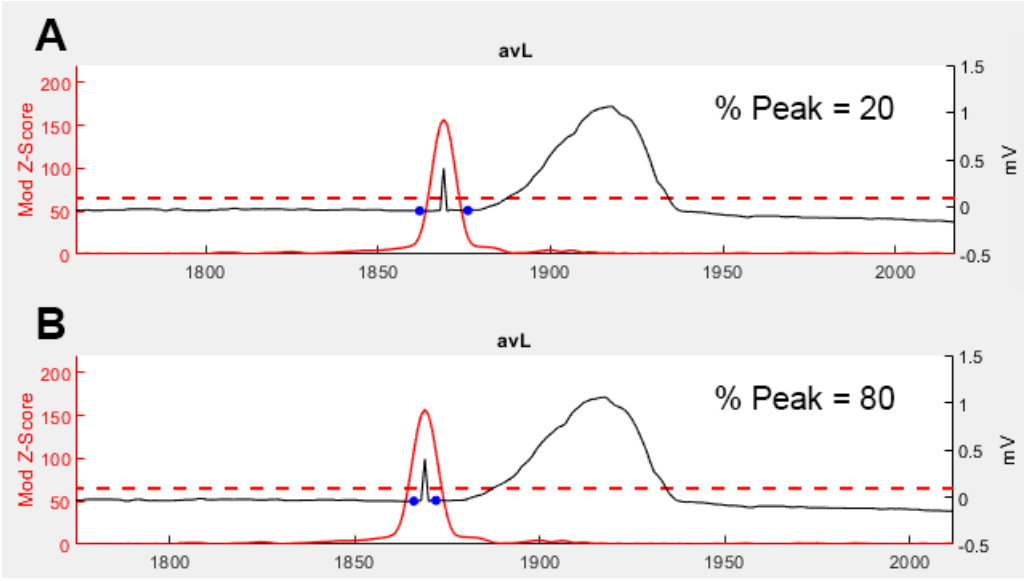


Figure 36

As the value of `pacer_zpk` increases, the width of the area of interpolation (between blue dots) is decreased.

GUI section 1.18 will also change depending on if the CWT method is active, if pacing is detected in at least 1 lead, and based on the value of `pacer_spike_num`:

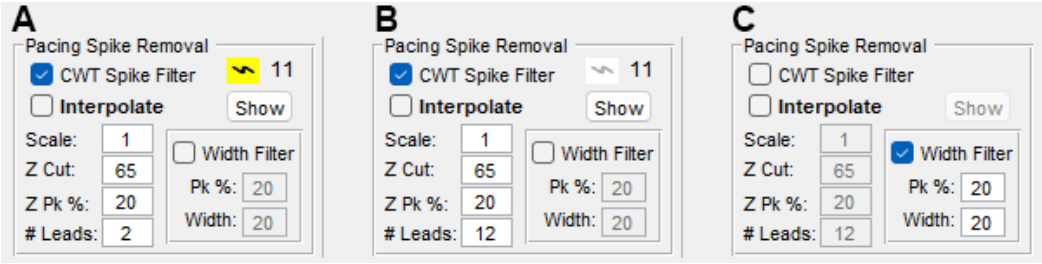


Figure 37: Changes in the GUI depending on how many leads had pacing detected. **A:** Pacing is detected in $\geq$ than `pacer_spike_num` leads. The number next to the electric bolt on a yellow background in the number of leads (out of 12) in which pacing was detected. **B:** Pacing is detected in $<$ `pacer_spike_num` leads. The number next to the electric bolt on a white background in the number of leads (out of 12) in which pacing was detected. In this case, `pacer_spike_num` was set to 12 but pacing spikes were only detected in 11 leads. **C:** CWT method is not being used or pacing is not detected. There is no number of leads displayed and no electric bolt displayed.

15.3 Baseline Correction

ECG machines do not usually set the zero voltage reference at the isoelectric interval of each lead. Setting the zero voltage reference correctly on an ECG lead is critically important to accurate measurements of the area under an ECG/VCG lead. The true isolectric (zero voltage) part of an ECG/VCG signal is the segment between the end of ventricular repolarization (T_{off}),

and the start of the P wave (TP segment). The importance of setting the zero voltage reference appropriately can be seen in this figure, where incorrectly setting the zero voltage can result is very large differences in measured areas:

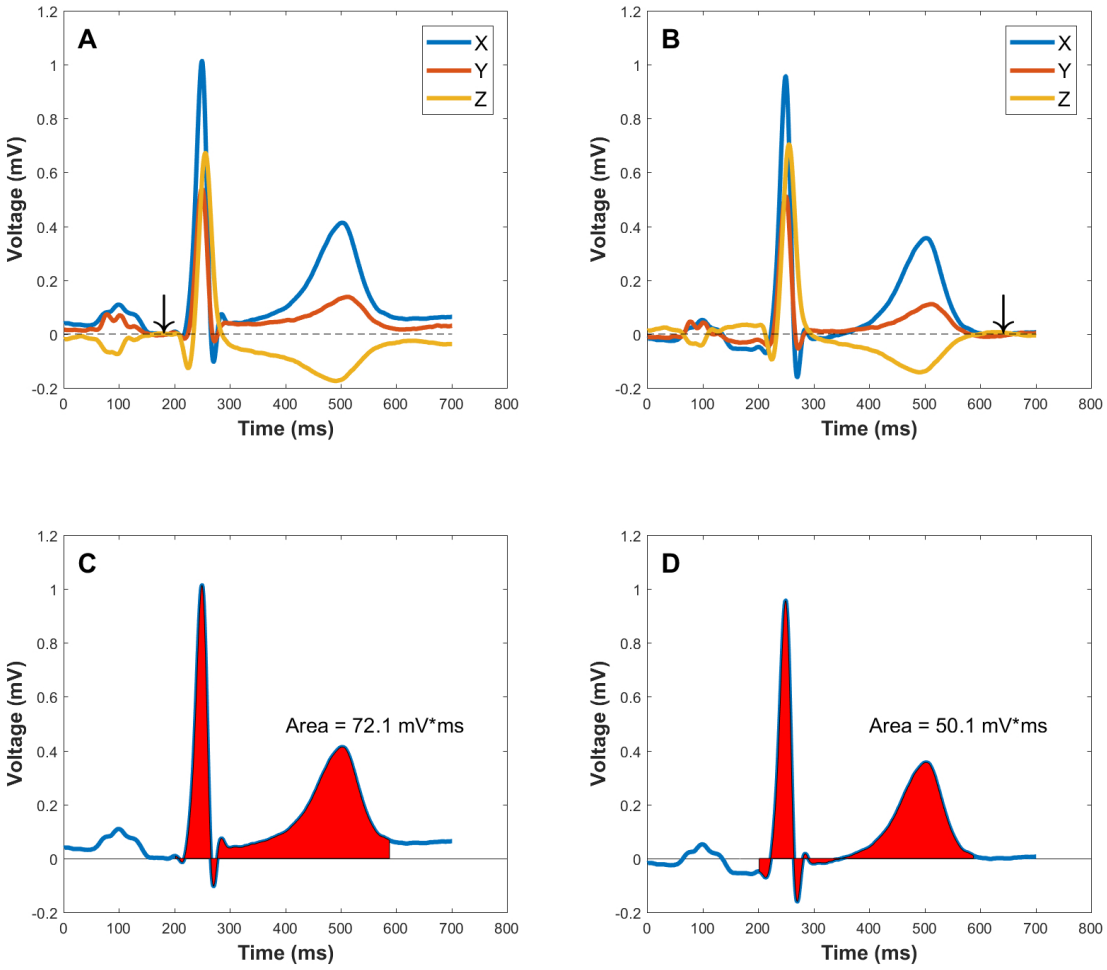


Figure 38: Significant differences in QRST area calculations due to different definitions of the vectorcardiographic origin point. **A**: Median VCG beats (X, Y, Z) without any baseline processing where the onset of the QRS complex (arrow) is defined as zero voltage. **B**: After baseline correction using the method described in the text, the TP segment, a physiologically isoelectric interval is now set as the zero voltage (arrow). **C** and **D**: Physiological baseline correction results in a 30% decrease in area under the X median beat. Reproduced with permission from [10].

High-pass filtering does remove DC offsets, but after filtering, the TP segment is not necessarily going to be set to approximate zero voltage. BRAVEHEART baseline correction adds a constant to the ECG lead such that the TP segment approximates zero voltage. This can be seen in **Figure 39** below, where high-pass filtering removes a significant amount of DC offset, but additional baseline correction is needed to set the TP segment as the zero voltage reference:

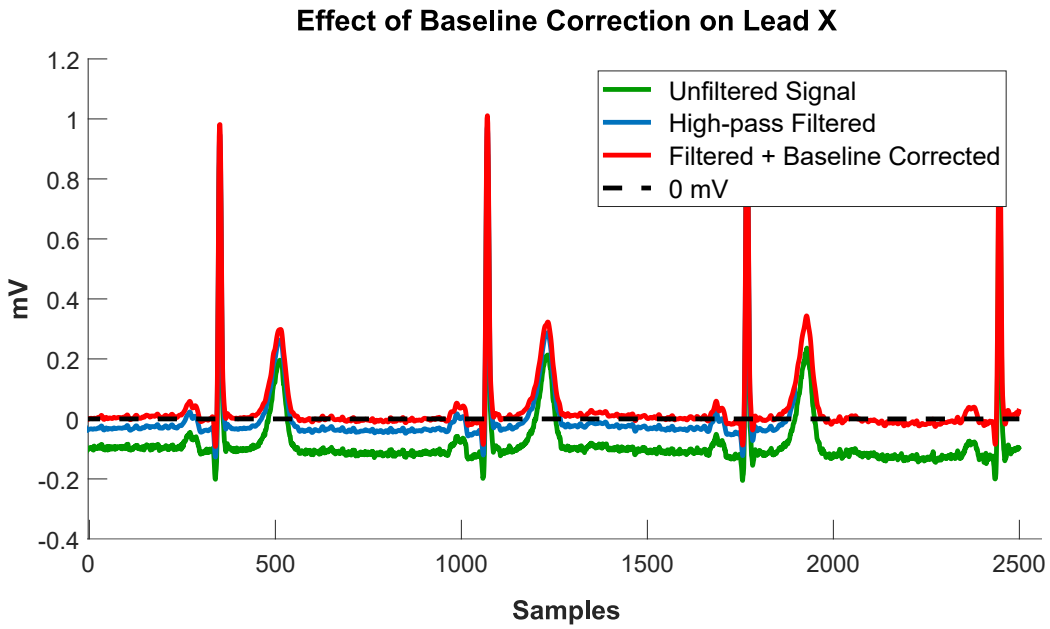


Figure 39: Effect of baseline correction after high-pass filtering. The green signal is unfiltered and has a large offset from the physiologic zero voltage. After high-pass filtering (blue), a large amount of this offset and low-frequency baseline wander is removed, but the TP segment, which is the true isoelectric interval of the ECG lead, is still not approximately zero voltage as the filters have no way to know to set the TP segment at 0 mV. After additional baseline correction (red), the TP segment approximates 0 mV (dashed black line).

Details of the procedure for baseline correction are available in the methods manuscript [1].

The effect of baseline offset correction can be seen by clicking on the Offsets button (GUI item 1.14). The original signals are shown in black, and the baseline corrected signals are shown in red. Two figures are generated – one shows all 16 leads and their basline corrections, and the other is a zoomed in view of the VM lead. An example of the offsets figure is shown here:

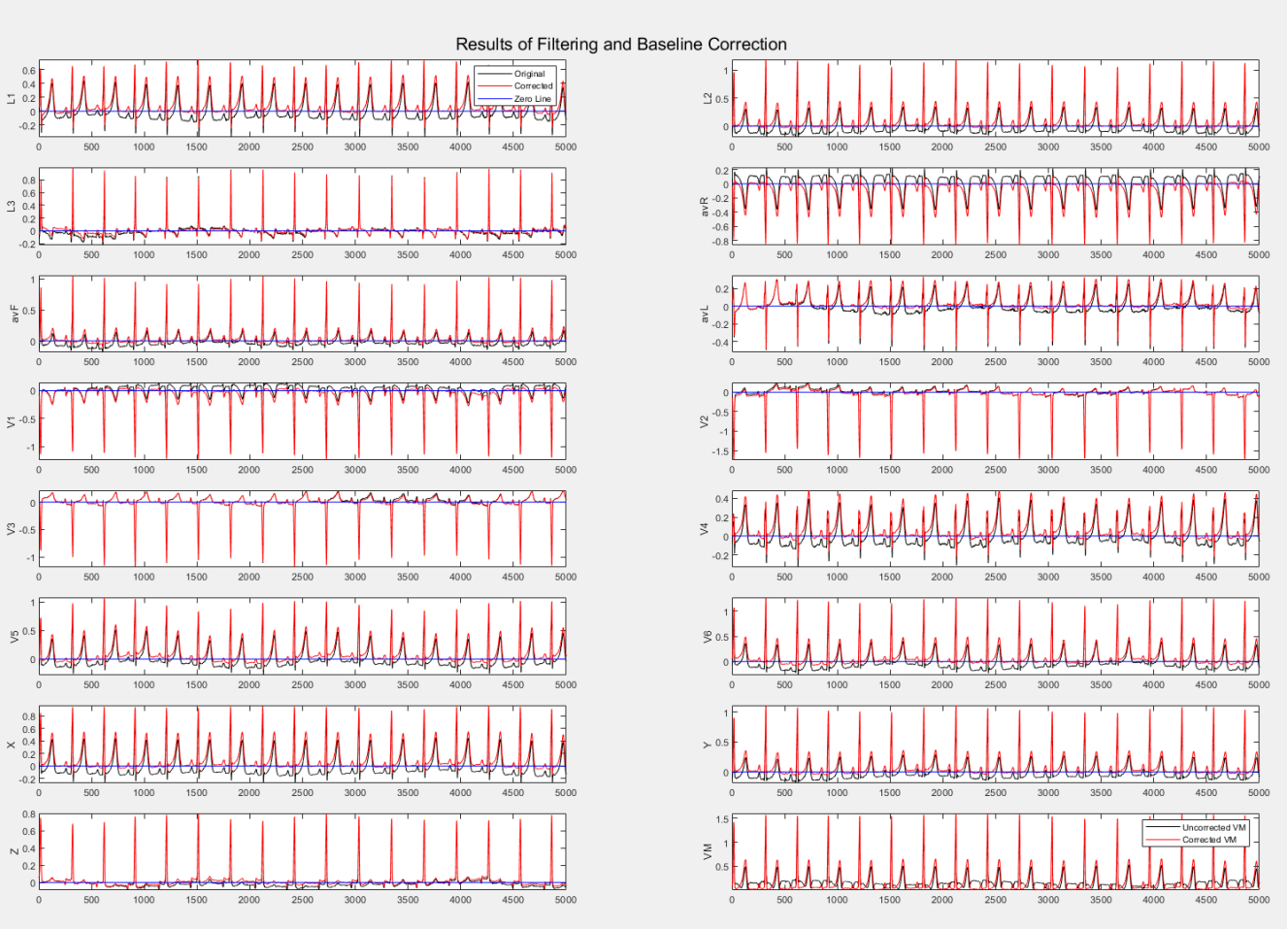


Figure 40: Effect of filtering and baseline correction on an ECG with significant baseline DC offset. The original signals are shown in black, and the filtered and baseline corrected signals are shown in red. The zero voltage line is shown in blue. Note that especially in leads I (L1), II (L2), V1, and X, there is significant deviation from zero baseline originally, and after filtering and baseline correction the TP segments now approximate zero votage. The vector magnitude (VM) lead is significantly distorted (black signal in the bottom right) without appropriate baseline correction as inappropriate DC offsets in the X, Y, and Z leads are amplified when calculating the VM = $\sqrt{X^2 + Y^2 + Z^2}$.

15.4 PVC and Non-Dominant Beat Morphology Detection

Accurate median beat construction and measurements depend on having a single QRST morphology among the beats that make up the median beat. Although median beat construction can deal with a few beats with disparate QRST morphology as long as these beats make up a low percentage of the total beats, results are more accurate if non-dominant beats such as PVCs or aberrant beats are excluded from analysis prior to median beat creation. In more extreme cases such as ventricular bigeminy with equal numbers of PVCs and normal QRST beats, if PVCs are not removed from analysis, the median beat will be a superposition of the normal QRST complex and the PVC, and calculated results will be highly inaccurate. For this same reason, it is not possible to rely on comparing each individual beat morphology to the median

beat morphology in order to determine what beats are PVCs or other non-dominant beats that should be excluded. BRAVEHEART has a module to automatically detect (and remove) PVCs and other non-dominant beats (from this point on the term "PVC" will refer to PVCs or other non-dominant QRST morphologies which could be due to aberration or ventricular pacing).

A detailed technical discussion of the methods used for PVC detection is available in the methods manuscript [1]. In brief, PVC detection utilizes normalized cross correlation (NCC) and normalized root mean squared error (RMSE) to compare the QRST morpholgies between pairs of QRST complexes. All pairs of beats are compared with NCC and RMSE, and the beat that has the best correlation with the most of the other beats is chosen as the "dominant" QRST morphology template. Compared to this template, beats that are outside of limits set by the user in the parameter `pvcthresh` for NCC (nominally 95%) and the parameter `rmse_pvcthresh` for RMSE (nominally 0.1) in `Annoparams.m`, are flagged as PVCs. BRAVEHEART compares morphologies like this in the X, Y, and Z leads, and given that a PVC can look similar to a native beat in a single lead, if a PVC is detected in 2 out of the 3 orthogonal X, Y, and Z leads, the beat is flagged as a "PVC". The user can then set if PVCs should be removed, or if the non-PVC beats should be removed (see `keep_pvc` in **Chapter 6.2**). In this way it is possible to analyze PVCs rather than dominant beats, automatically. In an analysis of the BRAVEHEART PVC identification algorithm, using NCC = 95% and RMSE = 0.1, sensitivity was 98.2%, specificity was 99.5%, and F1 was 97.7% for PVC detection (see methods manuscript [1] for additional details).

PVC detection and removal are controlled by the following parameters in `Annoparams.m` which are described in detail in **Chapter 6.2**.

- `pvc_removal` which enables or disables automatic PVC removal.
- `pvcthresh` which sets the valve for NCC (range 0–1).
- `rmse_pvcthresh` which sets the value for RMSE.
- `keep_pvc` which determines if PVCs are removed and retained for analysis.

When the GUI is used to control PVC detection/removal, as shown in **Chapter 12.10**, values for the above annotation parameters can be set in GUI section 9. The GUI additionally has a function which is called by the PVC Data button (GUI item 9.8). This brings up a figure which summarizes PVC detection in the X, Y, and Z leads, and can be useful for troubleshooting if needed:

Figure 41: Summary of PVC detection algorithm generated by clicking the PVC Data button (GUI item **9.8**). NCC and RMSE are displayed relative to a reference beat which is the beat that has the best correlation with most of the other beats in the lead (in this case beat 1 in all 3 leads). This reference can vary by ECG lead. Beat 10 is identified as a PVC based on NCC below threshold (`pvcthresh = 0.95`) and RMSE above threshold (`rmse_pvcthresh = 0.1`) in all leads (red box), and is flagged as a PVC (red box in the lowest row, 'Final PVC Markers').

15.5 Outlier Beat Detection

After PVCs, aberrant beats, paced beats, or other abnormal QRST morphology beats are removed, there may still be some beats that are contaminated with noise that are detrimental to median beat creation and subsequent calculations. This can occur because the PVC detector algorithm cannot look at the entire QRST complex since beats are of different length, and the algorithm therefore looks primarily around the QRS peak (see the methods manuscript [1] for additional details). Some beats that pass PVC detection can still have significant artifact in

other parts of the QRST complex that prevent accurate annotation and alignment to create the median beat. BRAVEHEART attempts to deal with this issue through removal of these "outlier" beats.

A detailed technical description of the outlier identification algorithm is presented in the methods manuscript [1]. In brief, to identify outliers, BRAVEHEART looks at the QR and RT intervals of all remaining beats after first pass annotation and PVC removal. A modified Z-score is calculated for each beats' QR and RT intervals. Beats that have a modified Z-score for the QR or RT interval above a threshold set by `modz_cutoff` in `Annoparams.m` are identified as outliers and can then be automatically removed based on the value of `outlier_removal` in `Annoparams.m` (see **Chapter 6.2**).

When using the GUI, Section 9 (see **Chapter 12.10**) is used to control outlier detection/removal. The GUI additionally has a function which is called by the Outlier Data button. This brings up a figure which summarizes outlier detection and can be useful for troubleshooting if needed:

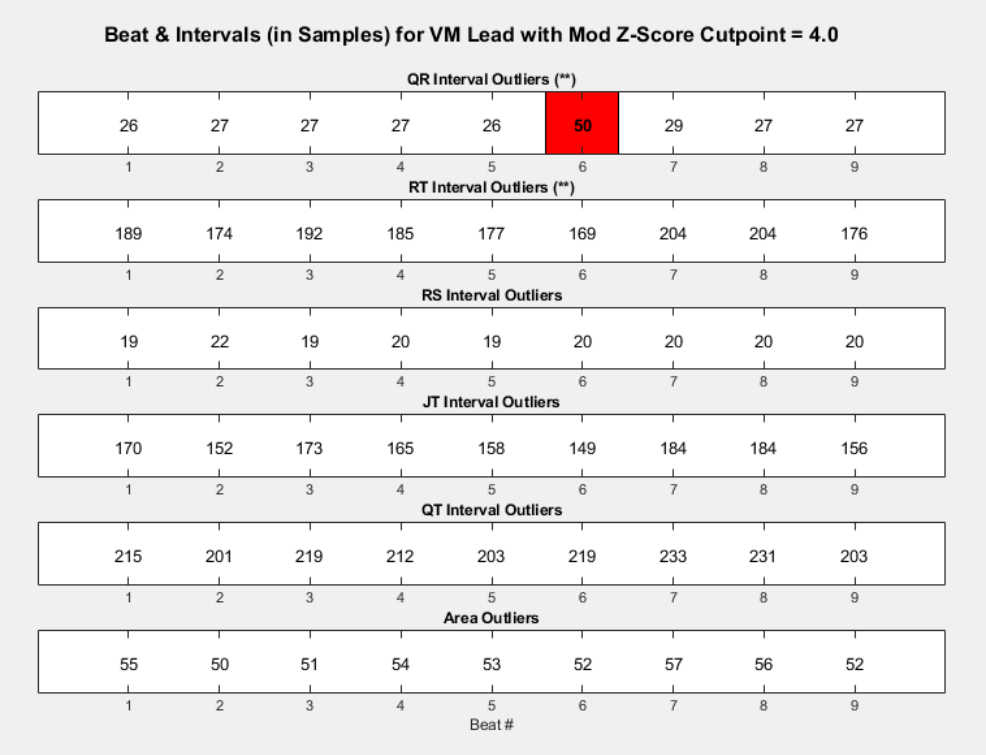


Figure 42: Summary of outlier detection algorithm generated by clicking the Outlier Data button in GUI item **9.2** for `example3.xml`. The first 2 rows display the QR and RT intervals for each beat, with each column representing a beat. QR or RT intervals that have a modified Z–score above the cutoff set in `Annoparams.m` are noted in red. In this example, beat 6 is classified as an outlier because the QR interval is out of range. Rows 3–6 can help diagnose issues with annotation, but are not used for classify a beat as an outlier, and beats with a modified Z–score above the set cutoff for rows 3–6 are displayed in black rather than red.

16 ECG Quality Assessment

BRAVEHEART was designed to allow efficient batch processing of large numbers of ECGs. The sample of the full batch of ECGs which are of overall poor quality, however, may not be known ahead of time, and with large ECG datasets it may not be feasible to manually review every processed ECG to ensure there is no significant artifact or other problems that may result in incorrect measurements reported by BRAVEHEART. To address this issue, BRAVEHEART has a module that assesses the quality of each processed ECG so that ECGs with the highest probability of being poor quality can be manually reviewed, while ECGs that are likely good quality without processing errors will not require manual review. All ECGs that trip the quality detector will not be of such poor quality that they need to be excluded from an analysis, but they likely should be reviewed to ensure they are of adequate quality. The quality assessment is designed to allow the user to easily customize the sensitivity and specificity of detection based on the quality of the ECG data being processed (e.g. data from a trial vs clinically acquired data), and quality detection can be completely disabled if wanted.

16.1 Quality Probability

BRAVEHEART quality assessment takes place in 2 ways. First we developed a logistic regression model where the output was ECG quality ("good quality" or "needs review") and the inputs were various features of the processed median ECG/VCG signals. Further details can be found in the methods manuscript [1]. After each ECG is processed, a probability (range 0–1) is reported in the BRAVEHEART output files as variable `quality_prob`, and as a percentage (range 0–100) in GUI item **4.5** ('Prob (%)') where values above a set threshold (see below) are shown in green indicating "good quality", while values shown in red indicate "needs review". The cutoff used to distinguish "good quality" and "needs review" ECGs can be adjusted to increase/decrease sensitivity or specificity as needed for a specific project. In general, a cutoff of ~80% tends to have a good mix of sensitivity and specificity, with both values >90%. Values >90% have higher specificity at the expense of lower sensitivity, and values <70% have higher sensitivity at the expense of lower specificity (see methods manuscript [1] for additional details). An example of ECG quality estimation using this regression method is shown here:

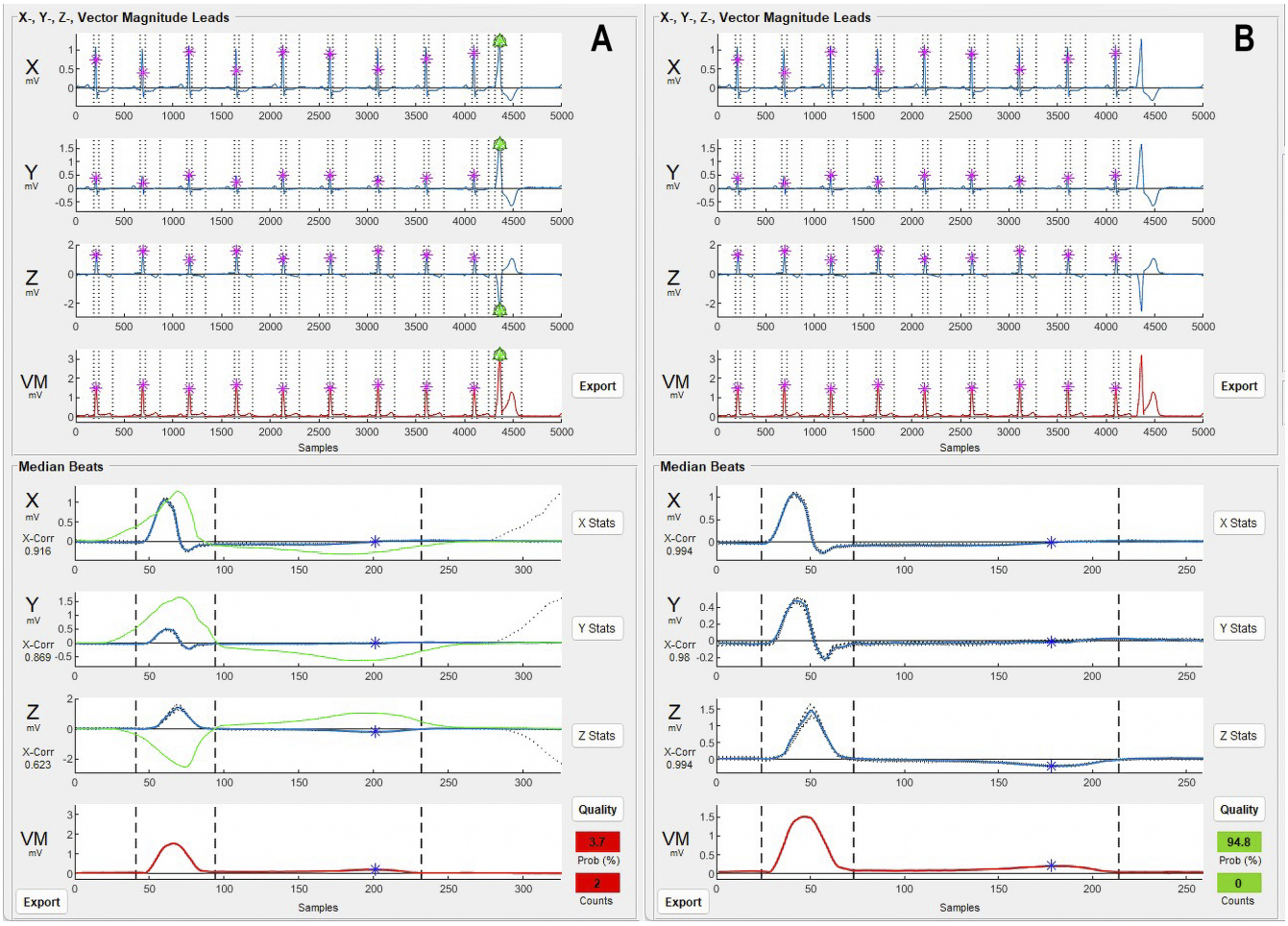


Figure 43: Example of changes in predicted quality in the setting of a PVC. **A:** Prior to PVC removal. The PVC (last beat) is highlighted in green. Because the PVC is not removed, the predicted probability of the median beat being "good quality" is low at 3.7%. **B:** Once the PVC is removed via automated PVC removal (see **Chapter 15.4**), the predicted probability of the median beat being "good quality" is now excellent at 94.8%. Additionally, in panel **A**, when the PVC is present, 2 quality flags (low predicted probability and low NCC) are triggered, while in **B**, after PVC removal, 0 quality flags are triggered (see **Chapter 16.2** below and **Figure 44**).

16.2 Quality Filters

The second way BRAVEHEART assesses quality is by using a set of filters that look at the results from ECG/VCG processing. Annotated median beats are checked for possible processing errors or missannotation by checking metrics including the heart rate, QT interval, QRS duration, number of beats included in the median beat construction, number of beats removed during PVC and outlier exclusion, T peak to QT ratio, T wave magnitude, average NCC between beats that make up the median beat, and low neural network annotation confidence. If any of these values are outside of a range which can be set by the user, the ECG is flagged for review.

16.3 Editing Quality Filtering Parameters - `Qualparams.m`

Note: In version 1.1.0 the method of adjusting quality filtering parameters was changed. The parameters are now editable via a MATLAB file `Qualparams.m` instead of editing the previous external file `quality_presets.csv`. The external file `quality_presets.csv` has also been renamed `Qualparams.csv` and is used only when running BRAVEHEART via a compiled executable when `Qualparams.m` cannot be edited.

Quality filtering is controlled by parameters in the file `Qualparams.m`. In cases where BRAVEHEART is being run via executable and `Qualparams.m` cannot be edited, the parameters can be set by editing `Qualparams.csv` which is an external file which contains the data contained within `Qualparams.m` in comma separated columns. When the program is run via executable, calls to pull data from `Qualparams.m` will instead pull data from `Qualparams.csv`.

The 12 `Qualparams.m` properties are listed here with their descriptions. Each of the 12 properties is assigned 2 numeric values as a vector enclosed in square brackets per MATLAB syntax. Each parameter's values are in the format `[min max]`, where the range of `min` to `max` is considered the "acceptable" range of values for that parameter. Therefore, values < `min` or > `max` will result in that quality flag being triggered.

```
qrs = [60 200];                 % Min/max range of QRS duration
qt = [250 700];                 % Min/max range of QT interval
tpqt = [0.5 Inf];               % Min/max range of T peak/QT ratio (nominal is min only)
t_mag = [0.05 Inf];             % Min/max range for T wave magnitude (nominal is min only)
hr = [30 150];                  % Min/max range for HR
num_beats = [4 Inf];            % # of beats left after PVC and outlier beats are removed
pct_beats_removed = [-Inf 60];  % of total number of beats removed to trigger
corr = [0.8 1];                 % Min/max range for average normalized cross correlation
baseline = [-Inf 0.1];          % Min/max range for baseline at the end of the T wave
hf_noise = [10 Inf];            % SNR for HF noise cutoff
lf_noise = [-Inf 0.02];         % mV for cutoff in variance in LF noise
prob = [0.8 1];                 % Logistic regression probability (range 0-1)
```

Note that some of the commented descriptions were shortened to fit on one line.

As shown below in **Chapter 16.4** and **Chapter 16.5**, 3 additional quality parameters, `missing_lead`, `NNet_prob_flag`, and `NNet_NaN` exist, but do not have user adjutable parameters.

You can use ± `Inf` to signify ±∞; `[0.5 Inf]` will only trigger a quality flag for values < 0.5. Similarly, `[-Inf 10]` will only trigger for values > 10. A filter can be completely disabled by setting the min/max values to `[-Inf Inf]`.

When using a compiled version of BRAVEHEART, `Qualparams.m` is not editable, and to adjust the quality filtering parameters, you can edit `Qualparams.csv`. This file contains the quality parameter in the first column, the minimum value in the second column, the maximum value in the third column, and a description of the parameter in the fourth column, with columns separated by commas:

```
qrs,                 60,    200,     Min/max range of QRS duration
qt,                  200,   700,     Min/max range of QT interval
tpqt,                0.5,   Inf,     Min/max range of T peak/QT ratio
t_mag,               0.05,  Inf,     Min/max range for T wave magnitude
hr,                  30,    150,     Min/max range for HR
num_beats,           4,     Inf,     # beats left after PVC and outlier removal
pct_beats_removed,   -Inf,  60,      % of total number of beats removed
corr,                0.8,   Inf,     Min/max range for average NCC
baseline,            -Inf,  0.1,     Min/max range for baseline at the end of the T wave
hf_noise,            10,    Inf,     SNR for HF noise cutoff
lf_noise,            -Inf,  0.02,    mV for cutoff in variance in LF noise
prob,                0.8,   Inf,     Logistic regression probability (range 0-1)
```

Brackets are not needed/used when editing `Qualparams.csv` like they are when editing `Qualparams.m`. When editing `Qualparams.csv` you can input -Inf and Inf. Blank values are loaded as $\pm\infty$ depending on which column they are in.

Adding new variables to the quality flagging can be done without too much difficulty, although doing so is beyond the scope of this guide.

16.4 Quality Output Files During Batch Processing

To facilitate review of ECGs that are potentially of poor quality during batch ECG processing, if any ECG is flagged for quality, a file named `check_ecg_list_<timestamp>.csv` is created which lists the ECGs which were flagged and the reason for flagging. The file contains columns which correspond to specific quality flags in `quality_presets.csv`:

- `qt_error` – `qt` (QT interval) is out of range.
- `qrs_error` – `qrs` (QRS duration) is out of range.
- `tpeakqt_error` – `tpqt` (ratio of T peak time to QT interval) is out of range.
- `tmag_error` – `t_mag` (T wave magnitude) is out of range.
- `HR_error` – `hr` (heart rate) is out of range.
- `num_beats_error` – `num_beats` (number of beats in the median beat) is out of range.
- `num_removed_beats_error` – `pct_beats_removed` (% of beats removed with PVC/outlier removal) is out of range.
- `median_sig_corr_error` – `corr` (NCC) is out of range.

- `baseline` – `baseline` (median VM voltage after T_{off}) is of out of range.
- `missing_lead` – A lead appears to be missing (Note: not editable in `Qualparams.m`).
- `hf_noise` – `hf_noise` (estimate of high-frequency noise) is out of range - see **Chapter 20.13**.
- `lf_noise` – `lf_noise` (estimate of low-frequency baseline wander) is out of range.
- `probability` – `probability` (predicted probability for "good quality") is out of range - see **Chapter 20.13**.
- `NNet_prob_flag` – `NNet_prob_flag` was activated due to low NN median beat annotation confidence (see **Chapter 19**). (Note: not editable in `Qualparams.m`).
- `NNet_NaN` – `NNet_NaN` was activated because the NN was unable to find all median beat fiducial points (see **Chapter 19**). (Note: not editable in `Qualparams.m`).

Values of `1` indicate that the specific quality metric was flagged, while values of `0` are normal.

If batch processing is set to save summary figures, BRAVEHEART also creates a separate directory of summary figures (folder named `figures_<timestamp>_check_ecg_list/`) only for the ECGs which are flagged for quality. In this way, it is easier to look through only the flagged ECGs rather than having to manually find flagged ECGs from within the full dataset.

16.5 Quality Output When Using the GUI

When the **Quality** button (GUI item 4.4) is pressed, it brings up a figure that shows which quality flags are normal in green (value = `0`), and which quality flags are abnormal in red (value = `1`) for the processed ECG:

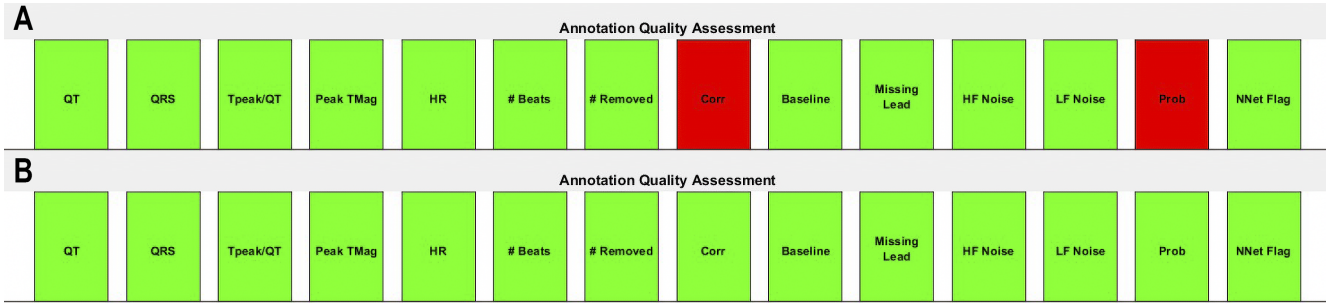


Figure 44: Example of changes in quality flags in the setting of a PVC as shown in **Figure 43**. **A:** Prior to PVC removal, 2 quality flags (low predicted probability and low NCC) are triggered, while in **B**, after PVC removal, 0 quality flags are triggered.

Note that the GUI combines the results of `NNet_prob_flag` and `NNet_NaN` into a single bar called "NNet Flag".

16.6 Other Uses for Quality Filtering

Although the quality filters are designed primarily for catching poor quality ECGs, they can also be used for other purposes including selecting ECGs from a large unlabeled dataset that meet certain morphologic criteria such as those with a certain heart rate range, prolonged QRS duration, or large T wave amplitude, among others.

17 Using `.anno` Files

`.anno` files have 2 main uses within BRAVEHEART. They are used to store the specific annotation parameters and beats that were used to process a specific ECG (these parameters are stored within the BRAVEHEART output data files as well), and to allow specific ECGs that don't conform to the standard parameter values within `Annoparams.m` or ECGs that require manual beat adjustment or removal, to be easily reprocessed within a batch of ECGs without having to manually adjust them over and over.

`.anno` files are `.csv` files that have the `.anno` extension to avoid issues with potential ECG file formats or results files that could also have the `.csv` extension. An example (and truncated) `.anno` file is shown below, and use of `.anno` files in the batch processing mode of BRAVHEART is shown in quick start Example 5 (**Chapter 5.5**).

Table 2: Example `.anno` file

```
275,                    299,   326,   502
925,                    949,   974,   1153
1533,                   1558,  1585,  1757
2160,                   2184,  2212,  2396
2768,                   2792,  2814,  2991
3370,                   3393,  3419,  3610
3991,                   4015,  4042,  4223
4626,                   4650,  4675,  4858
maxBPM,                 150,   ,
pkthresh,               95,    ,
lowpass,                1,     ,
highpass,               1,     ,
wavelet_level_lowpass,  2,     ,
wavelet_level_highpass, 10,    ,
wavelet_name_lowpass,   Sym4,  ,
wavelet_name_highpass,  db4,   ,
...
keep_pvc,               0,     ,
blanking_window_q,      0,     ,
blanking_window_t,      20,    ,
debug,                  0,     ,
```

The first n rows are the individual fiducial points for the n beats which were used to create the median beat. If ☐ Include Beats is not checked in the GUI, the beats are omitted. When `.anno` files are generated during batch processing of ECGs, the fiducial points are always included. The remaining rows starting with `maxBPM` correspond the all of the parameters in `Annoparams.m` and the parameter values that were used to process the ECG (see **Chapter 6**).

Note: `.anno` files do NOT save any manual edits you make on the median beat.

17.1 Generating `.anno` Files

There are 2 ways to generate an `.anno` file. In the GUI, item **3.14** includes an **Export** button to generate an `.anno` file for the ECG that is loaded. If the ☑ Include Beats checkbox is checked, the individual beats/fiducial points which make up the median beat are also saved in the `.anno` file. This can be useful if you had to manually remove a beat for some reason such as noise, if the automated outlier and PVC detection/removal was not effective, or other projects that involve looking at specific beats.

When performing batch ECG processing, checking the ☑ Annos box in the GUI, or setting `save_annotations = 1` in either `BRAVEHEART_batch.m` or `batch_settings.csv` when running via command line, will generate an `.anno` file, which includes beat fiducial points and annotation parameters, for each processed ECG. These files are stored in a folder called `annotations_<timestamp>/`. The annotation files within will have the same name as the processed ECG with the `.anno` extension.

17.2 Using `.anno` Files

An example of using `.anno` files with the GUI is shown in quick start Example 5 (**Chapter 5.5**). To use an `.anno` file within the GUI, first load the ECG as normal. Then click **Import** (GUI item **3.8**). A dialogue box will appear; choose the `.anno` file you want to load and click 'Open'. The current GUI settings are overwritten by the parameters in the `.anno` file, and if beat fiducial point annotations are included in the `.anno` file, only the included beats will be used to construct the median beat. ECG measurements are then calculated without needing to press the **Calculate** button. If you are processing other ECGs in the GUI after processing an ECG that included loading an `.anno` file, remember to reset the default annotation parameters with the Reset (GUI item **3.7**) before processing other ECGs.

When in batch processing mode via command line or GUI, BRAVEHEART checks the ECG file directory for any files with the same name as the ECG being processed with the file extension `.anno`. If an `.anno` file with the same name as the ECG being processed is present in the same directory as the ECG file, the ECG is processed using the annotation parameter values within the `.anno` file rather than `Annoparams.m`, and only using the beats specified in the file. In this way, if there is an ECG that requires a specific non-nominal values of a parameter such as `STend` to process properly, you can process the ECG and generate an `.anno` file. Edit the `.anno` file with the correct value of `STend` or whatever other parameters need to be adjusted.

Likewise, if outlier and PVC detection and removal are having an issue with a specific ECG and you want to manually exclude a specific beat, simply delete that beat from the `.anno` file. Then the next time you run a batch using BRAVEHEART with the same ECG (perhaps you want to run the batch of ECGs with a different transformation matrix or filtering parameters, or if you added a new parameter and want to calculate it on the same batch of ECGs), the different value of `STend` will load **for that ECG ONLY**, and the specific beat in question will be deleted. This can save time when running batch analyses multiple times.

18 First Pass Annotation

18.1 First Pass Annotation Algorithm

First pass annotation uses a heuristic algorithm to find Q_{on}, Q_{off}, and T_{off} for each QRST complex in the VM lead. The VM lead is used for first pass annotation because it is always positive and the QRS magnitude tends to be larger than the T wave magnitude which helps simplify calculations. First pass annotation is controlled with the following parameters in `Annoparams.m` (see **Chapter 6.2**):

- `autoMF`
- `autoMF_thresh`
- `MF_width`
- `QRwidth`
- `RSwidth`
- `STstart`
- `STend`
- `Tendstr`

First pass annotation is controlled by the `autoMFannotate.m` subroutine which uses these parameters, the locations of the dominant R peaks in the VM lead (see **Chapter 15.1**), and any pacing spikes (see **Chapter 5.4**), in order to set the search windows for the beginning and ending of the QRS complex and T wave. If `autoMF = 1` the QRS search algorithm dynamically estimates the QRS search window by estimating the the width of the QRS peak in the VM lead after median filtering with a filter width of `MF_width`. Using `autoMF = 1` is usually better than manually setting a static QRS search window using `QRwidth` and `RSwidth` because all beats are not necessarily the same width, and a fixed search window may not work for all beats. The search window is set for each individual peak to twice the the width of the region of points greater than the percentage `autoMF_thresh` of the height of the median-filtered dominant R-wave peak. If `autoMF = 0`, the provided, and static, values of `QRwidth` and `RSwidth` are used to set the QRS search window for all peaks.

Once the QRS search window is established, for each QRS complex, the start (Q_{on}) and end (Q_{off}) are determined as follows. First, the algorithm determines if there is an additional peak in the VM lead before the dominant R wave peak within a short distance (i.e. a Q wave). Two methods are then used to determine the start of the peak. Method A finds the maximum slope and walks out until it finds a point where the absolute value of the derivative is 2% of the maximum slope (or the smallest value found if not 2%). Method B looks for the first local minimum and labels this as the fiducial point. The point closest to dominant R-wave from

methods A and B is chosen as the fiducial point.

After Q_{on} and Q_{off} are located, a blanking window equal to `STstart` ms is set after the location of Q_{off} during which the T wave cannot be detected to avoid issues if Q_{off} is not detected correctly. At the end of this `STstart` window, the T wave search window starts and is extended by `STend` % of the mean RR interval. `STend` is set as a % of the mean RR interval rather than an absolute value in ms to minimize the need to change parameter values as QT interval is related to RR interval. The location of T_{off} is then detected by the method specified in `Tendstr`.

If `Tendstr` is `'baseline'` or `'tangent'`, the second half of the T-wave is fit to $A\exp\{(t/\sigma)^B\}$. The intersection of the function, or the tangent to the maximum negative slope with the baseline, is then calculated. If `Tendstr` is `'energy'`, the T-wave offset is chosen by a method based on the derivative of the signal (see reference [6]).

18.2 First Pass Annotation Figure

First pass annotation is summarized in the following figure:

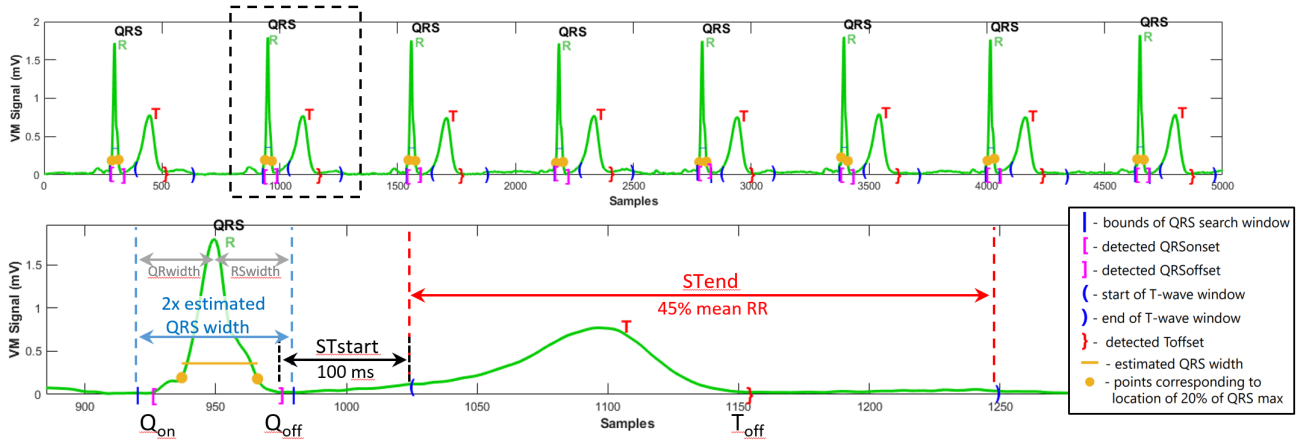


Figure 45: First pass annotation and search windows. The upper panel shows annotation search windows/detected points for the entire VM lead. The bottom panel shows a zoomed in view of the second beat (dashed box). If `auto_mf = 1`, the values of `QRwidth` and `RSwidth` (grey) are disabled and the QRS search window is dynamically estimated for each beat as twice the width of the median filtered QRS complex at `autoMF_thresh` % of the R peak voltage (nomimally 20% - yellow dots/line) for that beat centered on the R peak. If `auto_mf = 0` the values of `QRwidth` and `RSwidth` are used to set the width of the QRS search window before and after the R peak, respectively. The blue | symbols denote the beginning and end of the QRS search window and can help determine if the QRS detection window is too wide/narrow. Within the bounds of the QRS search window, Q_{on} and Q_{off} are detected as noted in the text with locations noted as [and], respectively. Once Q_{off} is detected, a blanking period set by `STstart` (nominally 100 ms) extends forward from the location of Q_{off}, marking the start of the T wave search window ((). The T wave search window extends forward by `STend` % of the mean RR interval (nominally 45%) and ends at the) symbol. Within the T wave search window bounds, the location of T_{off} is marked with a }.

18.3 Common Annotation Parameter Problems

The nominal value of `autoMF_thresh = 20`. Consider what can happen if `autoMF_thresh` is set too low:

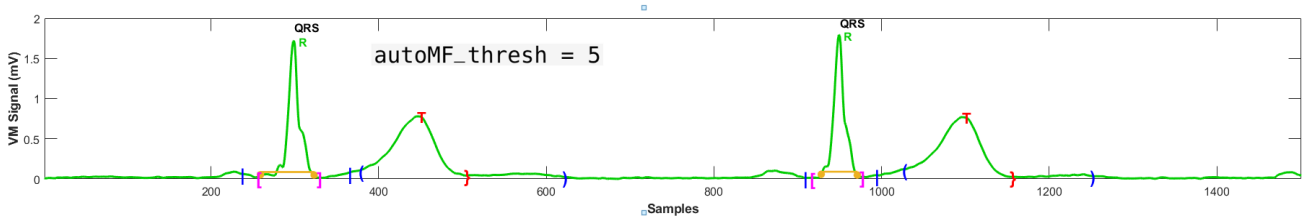


Figure 46: `autoMF_thresh = 5` is set too low. See text for details.

As shown in **Figure 46**, if `autoMF_thresh = 5` is set too low, more than the true QRS complex may be detected as part of the QRS complex (see orange circles and line), resulting in the estimated QRS width and the QRS search window (|) being too wide. This prevents accurate annotation of Q_{on} ([) and Q_{off} (]), but also can interfere with the T wave search window ((to)) and locating T_{off} (}) because the onset of the T wave search window starts `STstart` ms after the location of Q_{off}.

As shown in **Figure 47**, if `autoMF_thresh` is set too high, the QRS search window is likely to be too narrow:

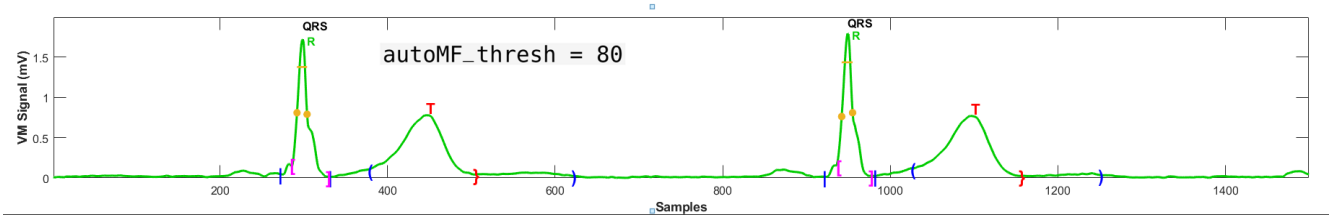


Figure 47: `autoMF_thresh = 80` is set too high. Se text for details.

Note that in this example, Q_{on} ([) is annotated later than the true QRS onset because the search window is not wide enough.

In cases where atypical QRST morphology is causing issues with first pass annotation, setting `autoMF = 0` in `Annoparams.m` or by unchecking the ☐ Auto Width checkbox, can allow manually setting the QRS search window by setting the values of `QRwidth` and `RSwidth`.

As shown in **Figure 48**, if `STend` is too short, the end of the T wave may not be found:

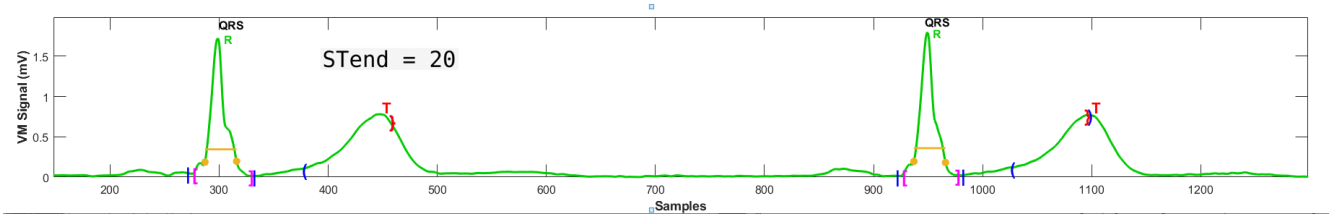


Figure 48: `STend = 20` is set too short. Se text for details.

In this example, the end of the T wave search window ()) is too short to include the entire T wave, and the location of T_{off} (}) is therefore not accurate. If `STend` is longer than necessary this usually does not cause as many issues as if `STend` is too short, as long as the end of the T wave search window does not encroach on the start of the next PQRST complex.

19 Neural Network Median Beat Annotation

Accurate annotation of the median beat is important for obtaining accurate measurements. We found that the heuristic methods which are used for first pass annotation were not accurate enough for median beat annotation, especially when QRST complexes had abnormal or very low amplitude T waves. This accuracy is less important for first pass annotation because first pass annotation needs to be just good enough to properly align each beat for median beat creation. We therefore trained and validated a custom bidirectional Long-Short Term Memory (LSTM) neural network (NN) designed specifically to annotate median VM beats. The technical details of NN training and validation are available in the methods manuscript [1].

The NN included with BRAVEHEART is highly accurate when compared to board certified cardiac electrophysiologists and its accuracy was tested in 2 separate datasets.

Testing set 1 was performed at the time of NN training using data that was not used for training or validation of the NN, and showed excellent agreement with ground truth, well within acceptable limits for computer based automatic fiducial point annotation [11, 12, 13, 14]. Testing set 2 was performed using a completely different dataset collected for different purposes. The NN also had an excellent accuracy with this second, independent dataset. Results are summarized here:

Table 3: Neural Network Annotation Accuracy

Result	**Q_{on}**	**Q_{off}**	**T_{off}**	**QRS duration**	**QT interval**
Set 1 Error Mean ± SD (ms)	-0.3 ± 3.9	-1.2 ± 5.1	-0.1 ± 7.0	-0.9 ± 6.2	0.2 ± 7.9
Set 2 Error Mean ± SD (ms)	-0.1 ± 3.3	-1.7 ± 4.0	1.2 ± 5.8	-1.6 ± 5.4	1.3 ± 6.7

As can be seen, the mean difference between NN predicted and ground truth fiducial point locations were $< \pm 2$ ms (1 sample at 500 Hz) for all points in both testing datasets. The methods manuscript [1] contains extensive information about NN architecture, training, and testing results.

The output of the NN can be visualized by viewing the 'Debug' figures enabled with the ☑ Debug checkbox (GUI Section 3 **Chapter 12**) and generated after pressing Calculate which shows the probability scores for each sample being either a "QRS complex", "T wave", or "other":

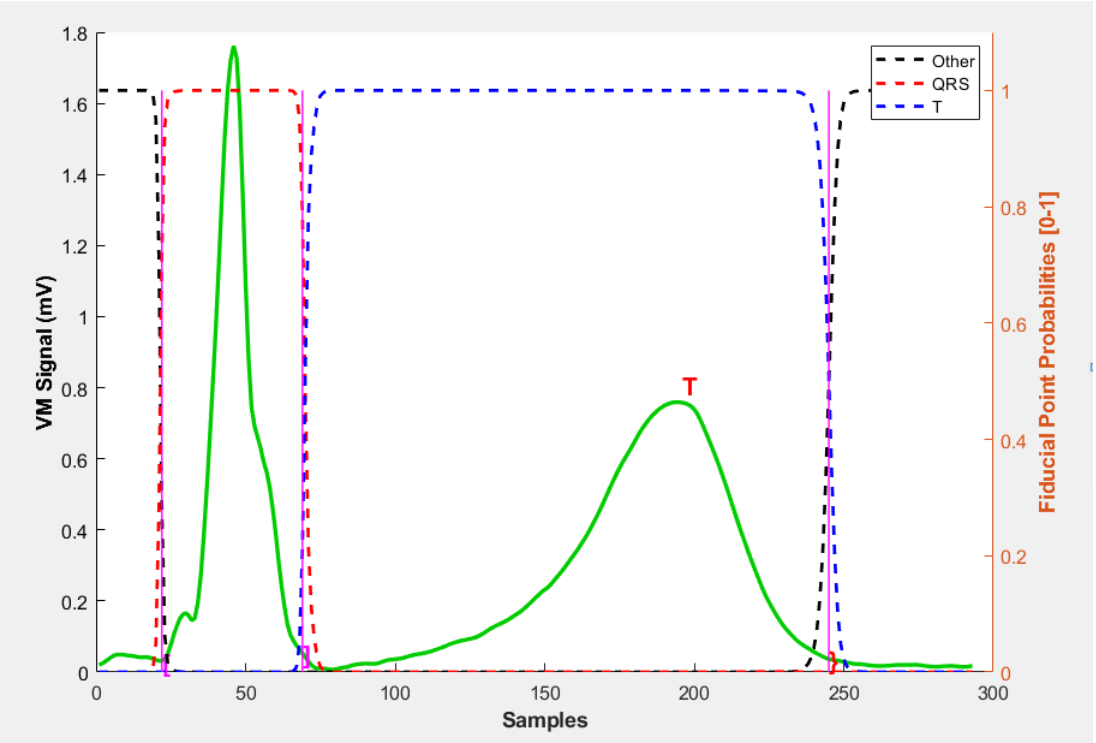


Figure 49: Debug figure for median beat annotation. The dashed lines are the NN predicted probabilities of a sample being a "QRS complex" (red), "T wave" (blue), or "other" (black). Each sample in the median beat signal is assigned the label that is the highest of the 3 probabilities. Q_{on} is set as the sample where "other" transitions to "QRS complex", Q_{off} is set as the sample where "QRS complex" transitions to "T wave", and T_{off} is set as the sample where "T wave" transitions to "other". Q_{on} is marked with a [, Q_{off} is marked with a], and T_{off} is marked with a } similarly to first pass debug figures. In rare cases where multiple potential fiducial points are detected, logic is used to select the best point – see the methods manuscript [1] for additional details.

20 Figures

BRAVEHEART allows visualization of ECG and VCG data in numerous ways. The following section describes the figures/visualizations that can be generated primarily by the BRAVEHEART GUI. When batch processing, normally only summary figures (**Figure 50** and **Figure 51**) are generated, although with code modification it is possible to generate any of the figures in this section during batch ECG processing.

20.1 Summary Figure

The summary figure is generated by batch processing for each ECG that is processed if `save_figures = 1` or when ☑ Figs is checked (GUI item **1.19**).

When using the GUI to analyze individual ECGs, the summary figure can be generated using the **Summary** button in GUI item **7.1**. To save the summary figure as a `.png` file click the ☑ Save checkbox (GUI item **7.3**) prior to clicking **Summary**. Two examples and descriptions of the Summary Figure are shown here:

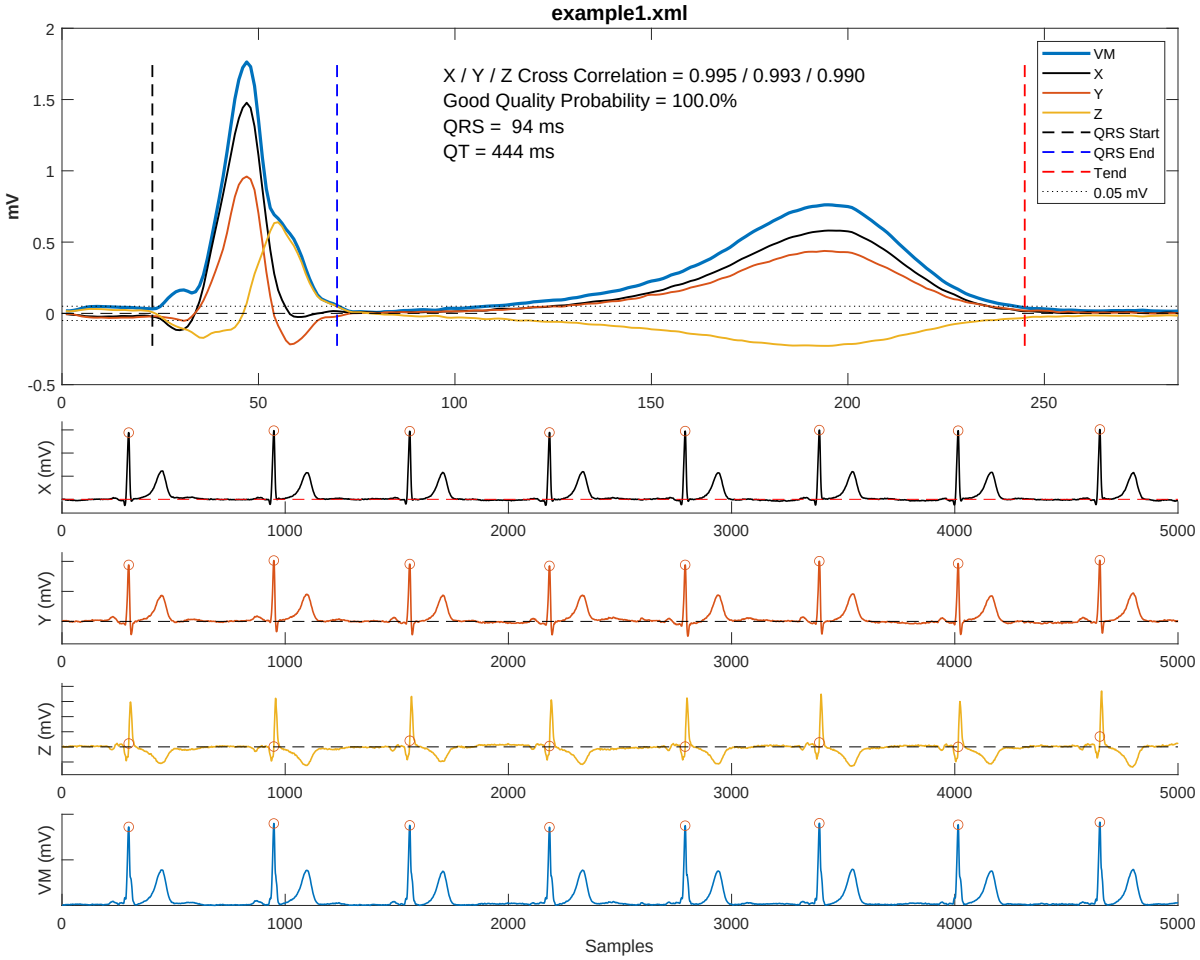


Figure 50: Summary figure for `example1.xml` generated when `save_figures = 1` when set in `braveheart_batch.m`, when ☑ Figs is checked prior to running batch ECG processing from the BRAVEHEART GUI, or when clicking **Summary** in the '**View/Save ECGs**' section of the GUI after ECG/VCG processing. The summary figure shows the median X, Y, Z, and VM beats in the upper half, and the individual beats used to create the median beats in the lower half. Beats that are included in the median beat are marked with an orange circle on the VM R peak. Beats that are not included do not have an orange circle on the VM R peak. Beats that are removed due to PVC or outlier detection are marked with 'PVC' or 'Out', respectively. Beats that do not have an orange circle and were not annotated as either a PVC or outlier were removed either due to being too close to the start/end of the ECG signal, due to manual beat removal, or due to being missed by initial QRS detection (below threshold or within the blanking period). The summary figure is useful to give you a quick view of which beats are included, which beats were excluded and why, the quality of median beat creation (via cross correlation values and probability of the ECG being "good" quality) at the top, and median beat intervals and annotations.

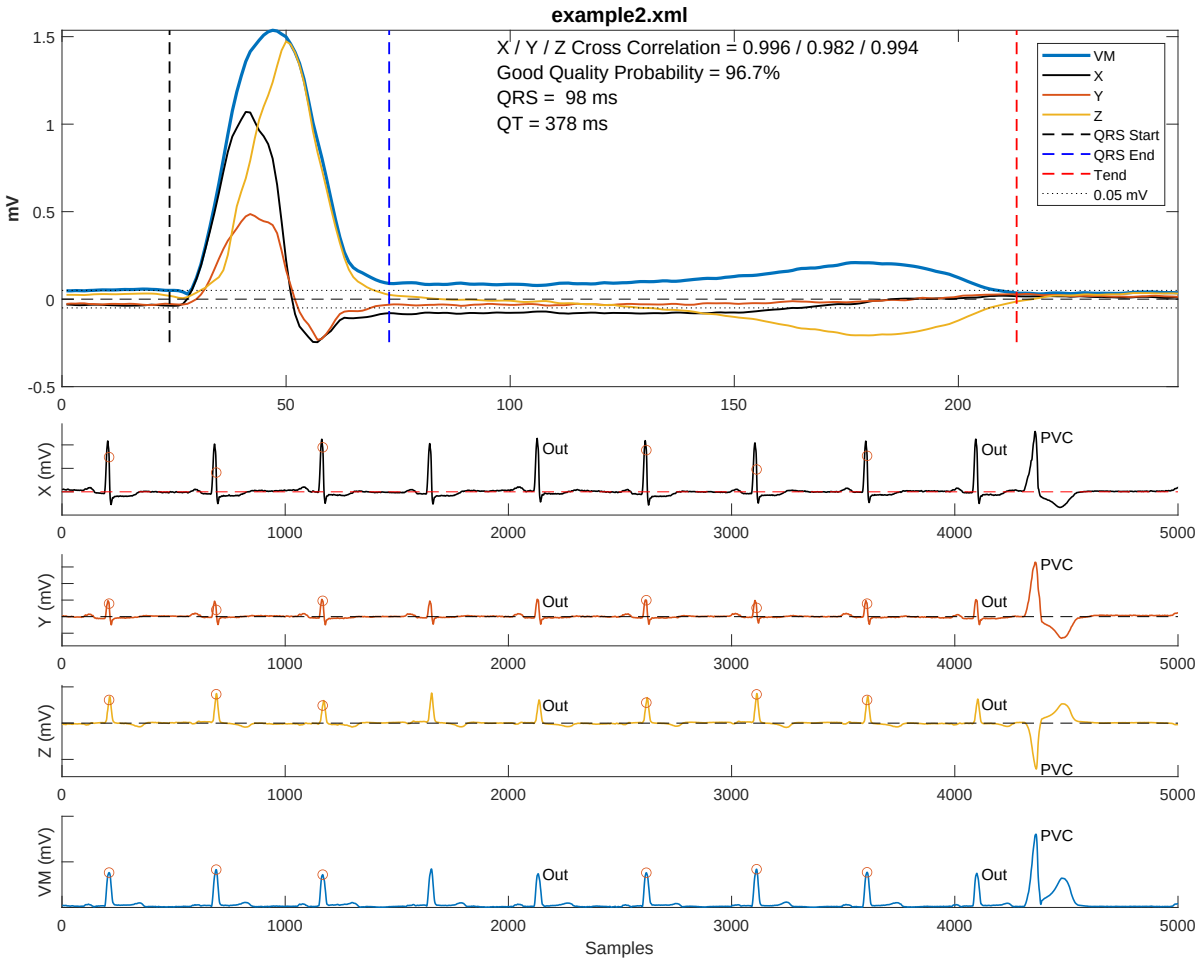


Figure 51: Summary figure for `example2.xml`. The figure layout is the same as noted in **Figure 50**. In this example the PVC has been removed by automatic PVC detection and removal. Note that the PVC does not have an orange circle on the VM R peak, indicating the the beat is not included in the median beat. The PVC is labeled as "PVC" indicating that the beat was removed with PVC detection/removal. Beats that were detected and removed as outliers are labeled as "Out". For illustrative purposes beat 4 has also been manually removed and therefore does not have an orange circle on the VM R peak.

20.2 12-Lead ECG/VCG

Once an ECG is loaded, the 12-Lead ECG and/or VCG can be displayed with the (`12L ECG`) and VCG (`X,Y,Z VCG`) buttons in GUI item **7.1**. The inclusion of major/minor grid lines can be enabled/disabled by choosing an option in dropdown in GUI item **7.2**. To save the ECG/VCG as a `.png` file, click the **`Save .png`** button after loading the figure. The 12-lead ECG and VCG for `example1.xml` are shown here:

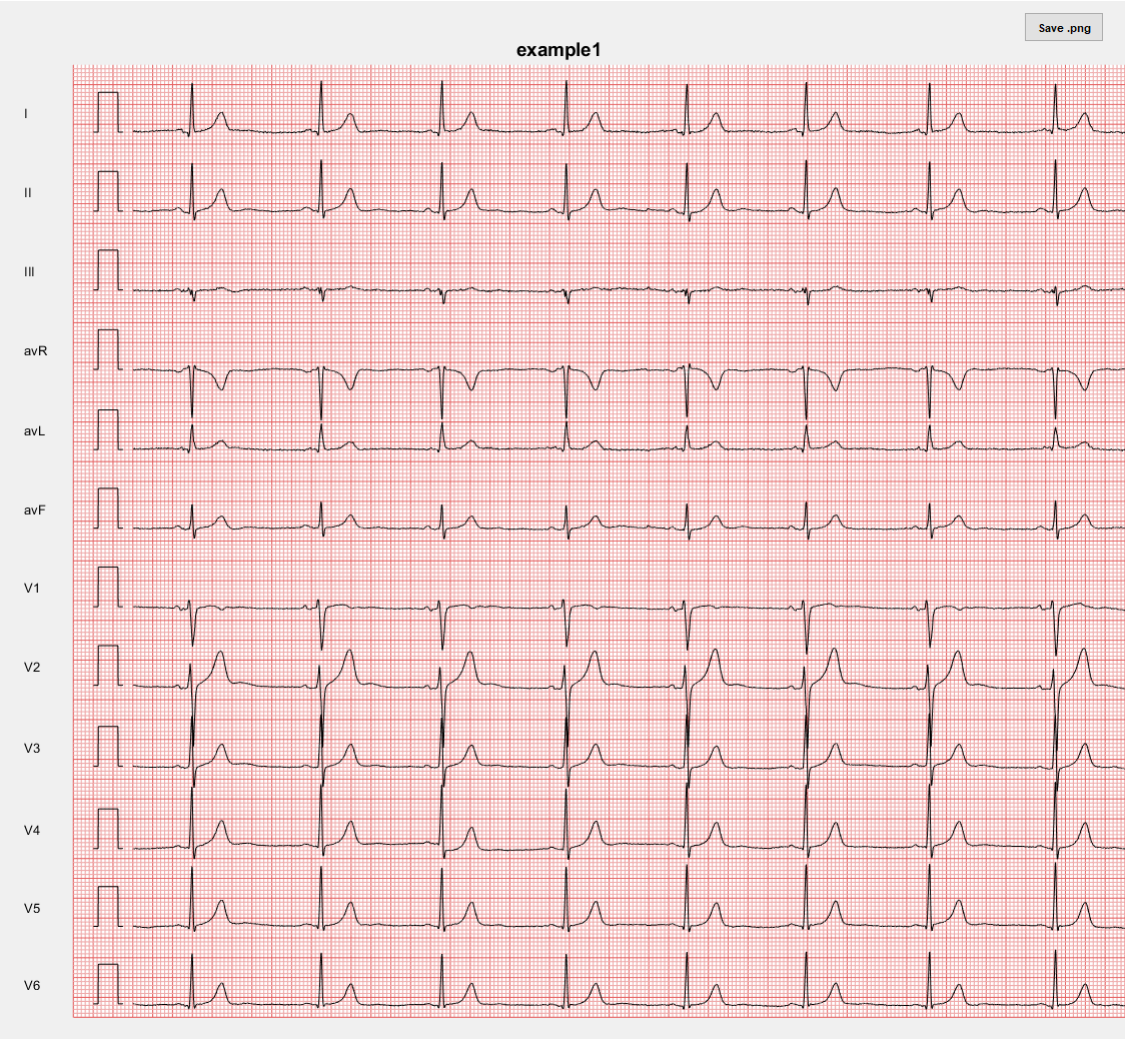


Figure 52: 12-lead rhythm strips for `example1.xml`.

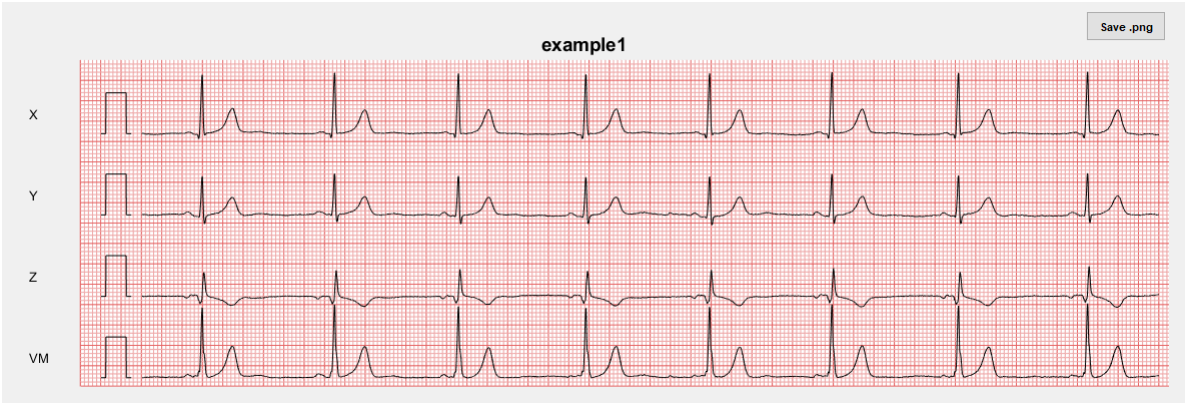


Figure 53: VCG and VM rhythm strips for `example1.xml`.

20.3 Median Beats

The 16 median beats can be displayed by clicking the Med. Beats button in GUI item **7.1**. The inclusion of major/minor grid lines can be enabled/disabled by choosing an option in the dropdown in GUI item **7.2**. To save the median beat figure as a `.png` file, click the **Save .png** button after loading the figure. The median beat figure for `example1.xml` is shown here:

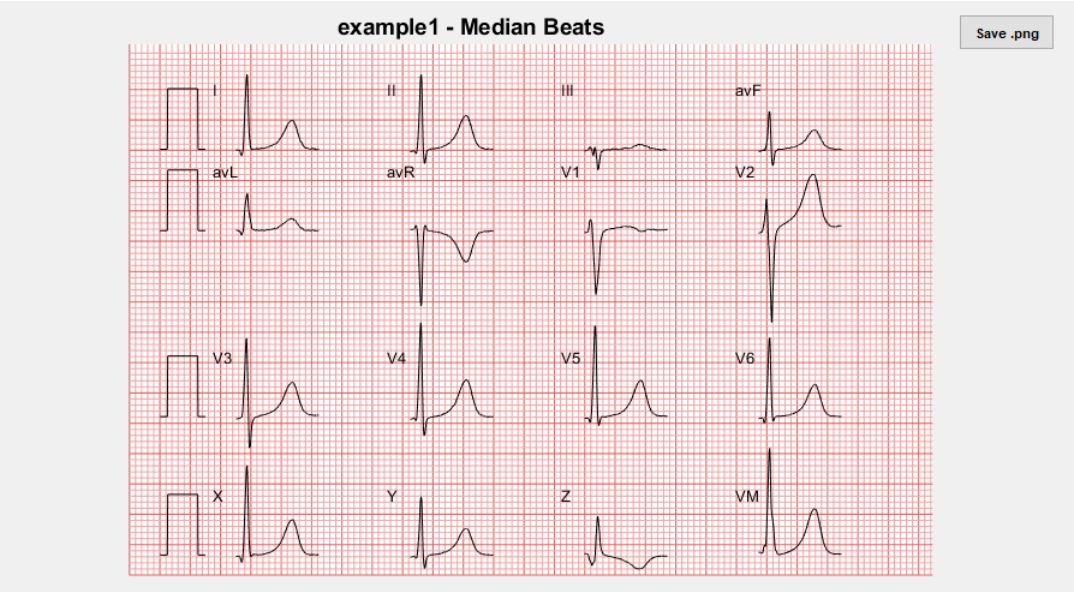


Figure 54: 16 median beats for `example1.xml`.

Detailed measurements of the median beats can be shown by clicking the **Lead Morphology** button (GUI item **5.3**). See **Figure 58** for additional information.

20.4 Normal Values

BRAVEHEART can display age, sex, BMI, and race specific normal ranges for certain parameters (see **Chapter 22**), and show if calculated values are within normal range by clicking on the Normal Vals. button. Values in black are within normal range, and values in red are outside the normal range. The parameters shown in this figure are the same parameters that will have ↑ or ↓ next to them in the '**VCG Calculations**' section (GUI item **6.2.1**) to indicate if the value is above or below the normal range, respectively. To save the normal values figure as a `.png` file, click the **Save .png** button after loading the figure. Further details on normal values/ranges can be found in **Chapter 22**. The normal values figure for `example1.xml` is shown here:

Figure 55: VCG calculations and their relationship with age, sex, BMI, and race specific normal values for `example1.xml`. The normal ranges are shown in yellow, with the upper and lower bounds shown with grey text. In this example, SVG X, SVG Y, SVG magnitude, SAI VM, and SAI QRST are out of range (red lines and text), while the other parameters which are within the normal range with black lines and black text.

20.5 VCG Loops

The VCG loops figure displays the X, Y, and Z median beats and the associated VCG loops displayed in orthogonal views, and can be generated by clicking the VCG Loops button. To save the VCG loops figure as a `.png` file, click the **Save .png** button after loading the figure. The direction of QRS propagation is shown via color scale (red to blue). This can be toggled off/on by clicking the **Toggle Color** button. The VCG Loop figure for `example1.xml` with and without propagation is shown here:

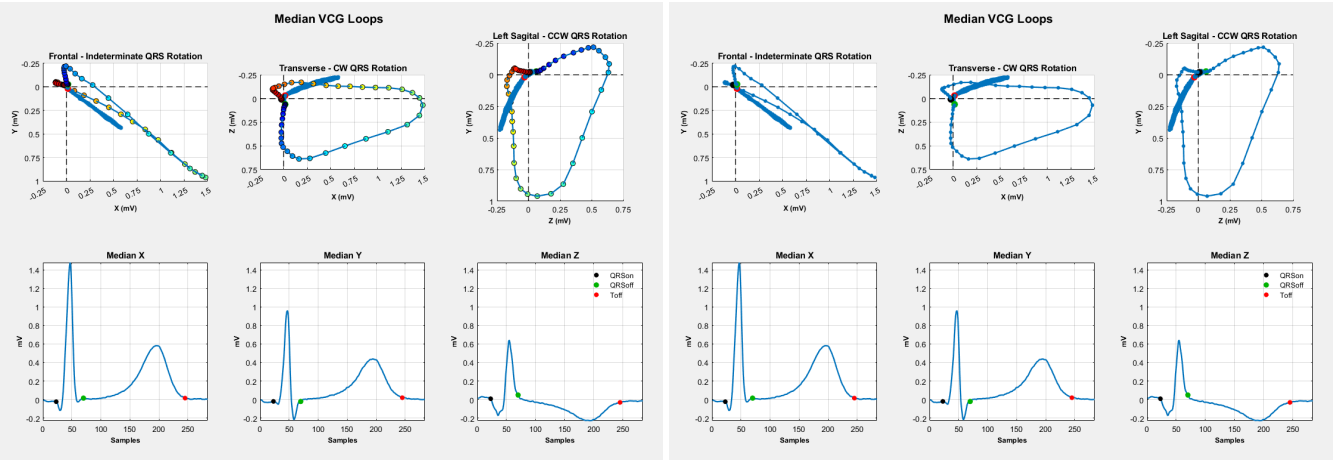


Figure 56: VCG loop figure for `example1.xml` with (left) and without (right) QRS propagation.

20.6 3D VCG

The 3D VCG is displayed in GUI section **6.1** by clicking on the **Vectorcardiogram** button (GUI item **5.1**). This figure can be rotated by using the rotation tool (GUI item **10.1**), and options in GUI item **6.1.1** can be used to change what information is displayed in the figure (see **Chapter 12.7**). To save the 3D VCG figure in the current orientation, click on the Save button in the lower right of the figure (GUI item **6.2.8**).

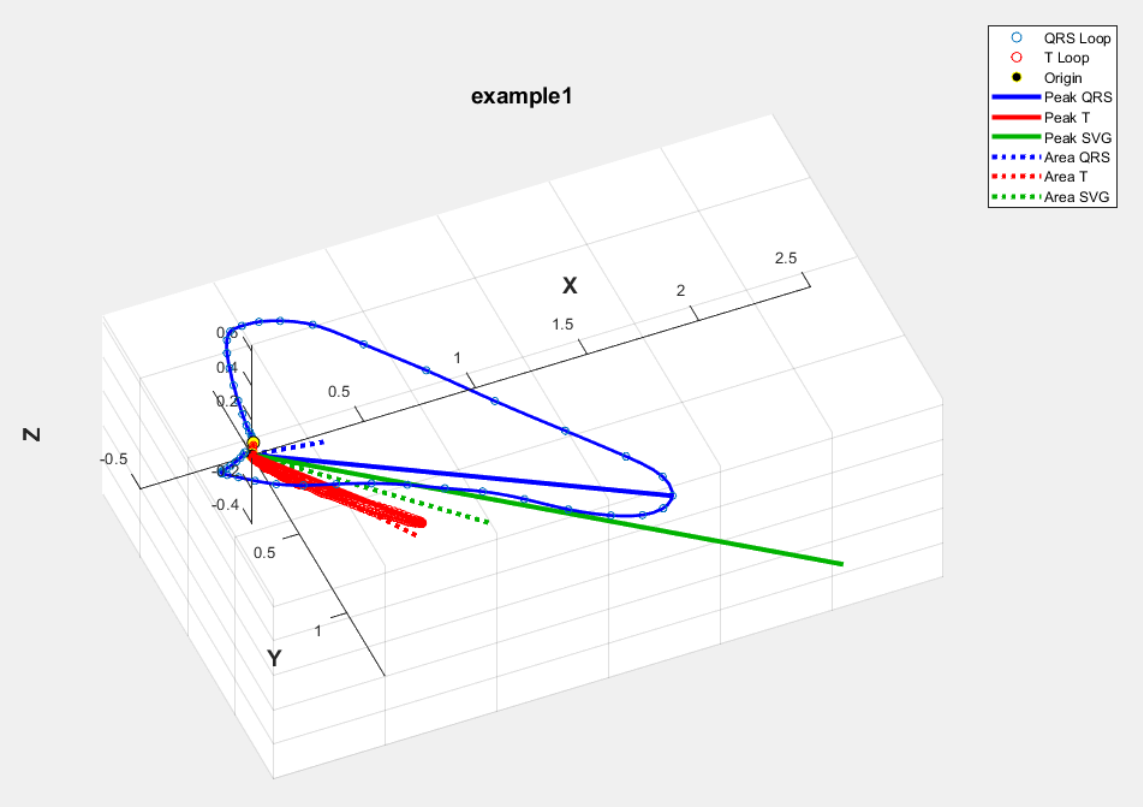


Figure 57: 3D VCG as displayed in GUI item **6.1**. The QRS loop is shown in blue and the T wave loop is shown in red. In this specific situation various vectors are shown as described in the legend. Options for VCG display are shown in GUI item **6.1.1**

20.7 Lead Morphology

The **Lead Morphology** button (GUI item **5.3**) displays a pop-out window with all 16 median beats, fiducial points, and the measurements of the R, S, and T waves. This displays the majority of results contained in the `Lead_Morphology` results class. The lead morphology figure for `example1.xml` is shown here:

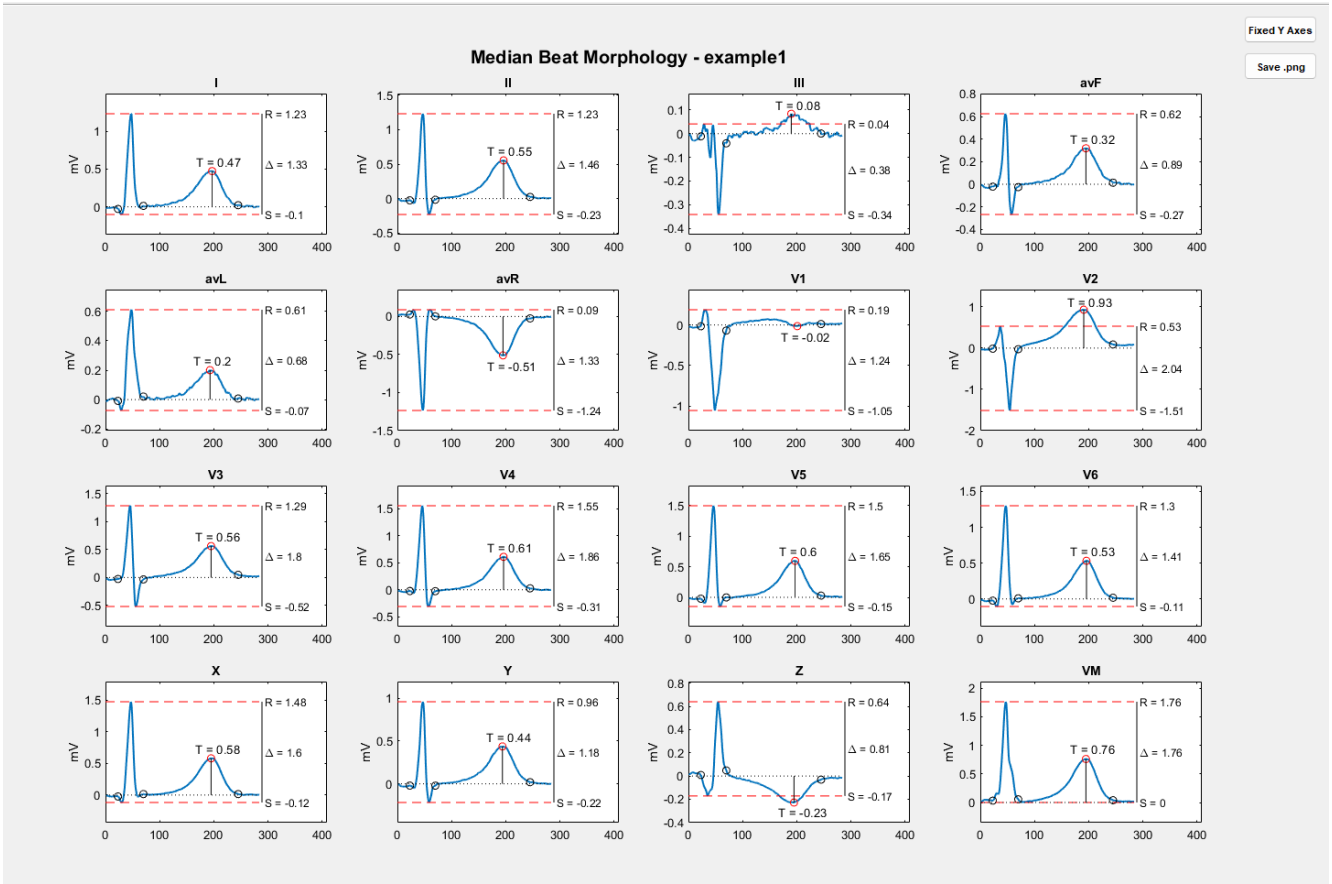


Figure 58: Measurements on 16 median beats. Values are in mV.

The buttons at the upper right of the figure allow the user to save the figure as a .png or to adjust the Y-axes so that all beats use the same Y-axis spacing. By default, the Y axis is adjusted to maximize display of the lead.

Clicking on Fixed Y Axes allows visualization of all leads on the same Y axis scale:

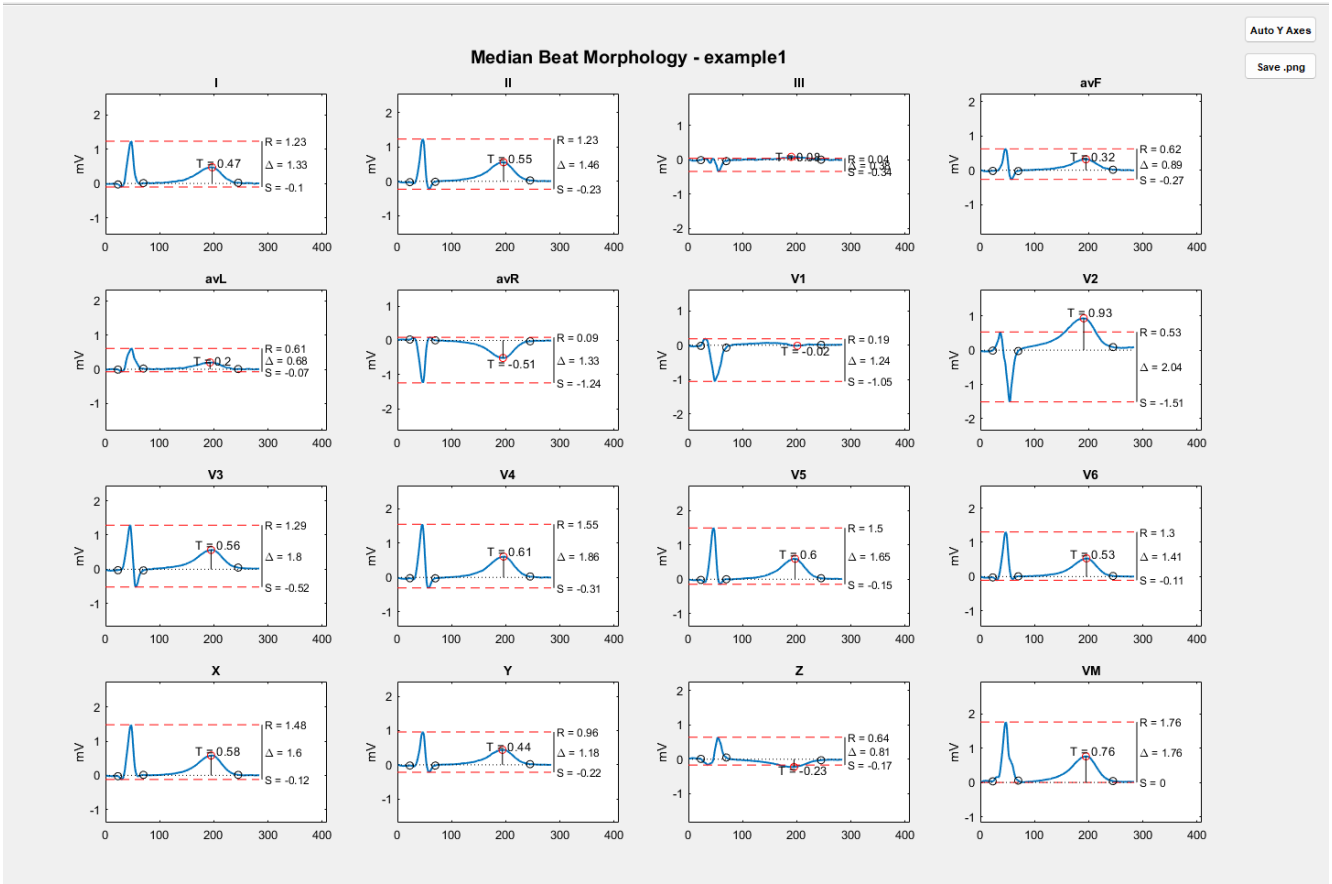


Figure 59: Measurements on 16 median beats with all beats using the same Y axis scale. Values are in mV.

Clicking on Auto Y Axes restores the original scaling for the Y axes.

In addition to the above figure, clicking on the **Lead Morphology** button displays a pop out figure with all leads overlapping:

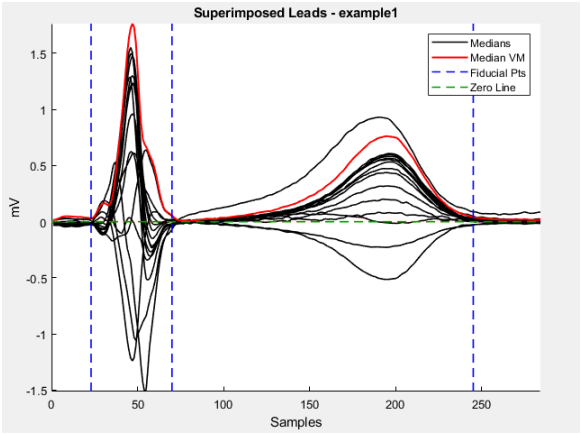


Figure 60: 16 median beats superimposed with fiducial points.

and a pop out figure with the frontal plane QRS axis:

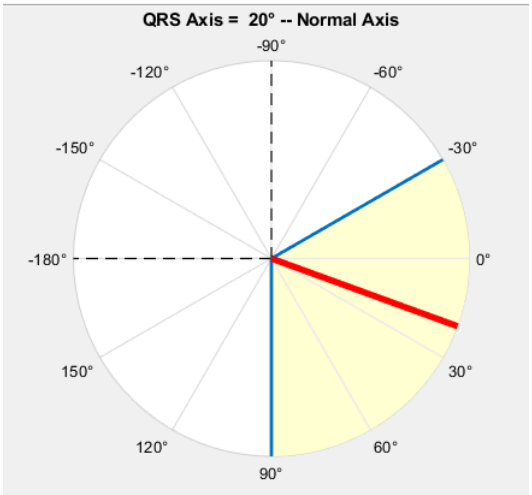


Figure 61: Frontal plane QRS axis The normal range is shaded in yellow..

20.8 VCG Speed

The VCG speed figure is displayed by clicking on the **VCG Speed** button (GUI item **5.5**) and then the **Graph** button (GUI item **6.5**). The speed graph does not automatically display when an ECG is processed. The **Popout** button opens the speed figure displayed in GUI item **6.5.10** in a new window to more easily adjust the figure. The X-axis is milliseconds after Q_{on}. For further details see **Chapter 12.7**. The VCG speed figure for `example1.xml` is shown here:

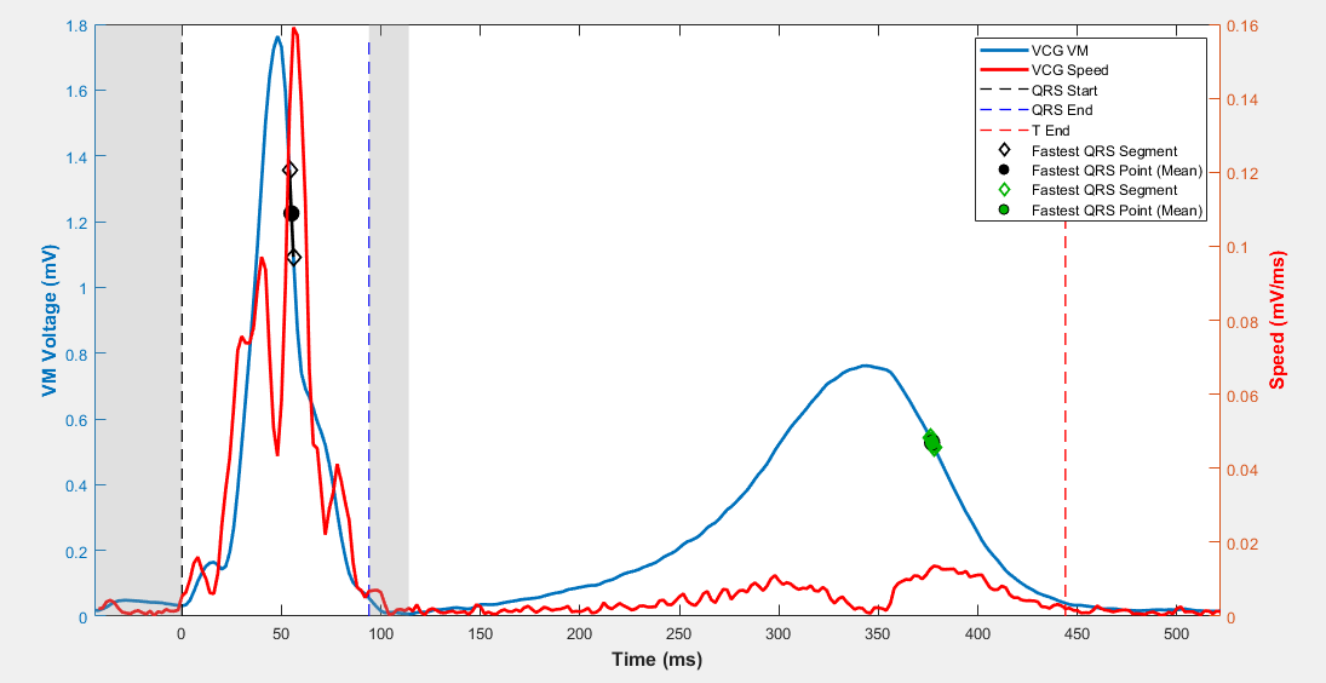


Figure 62: VCG Speed Figure.

20.9 Animated VCG

An animated VCG can be generated by clicking the Animate button (GUI item **6.1.3**). The 'Step' textbox sets the number of samples that are advanced per frame.

20.10 X-, Y-, Z- Stats Figures

The X-, Y-, and Z- stats figures display information on the beats that make up the median X, median Y, and median Z beats, respectively. These figures are generated by clicking on the X Stats, Y Stats, or Z Stats buttons (GUI item **6.4.3**). Beat-by-beat QRS duration, QT interval, QRST area (SVG), absolute QRST area (SAI), and the individual beats (black tracings) that make up the median beat (green tracing) are shown with normalized cross correlation (NCC). The X-stats figure for `example1.xml` is shown here:

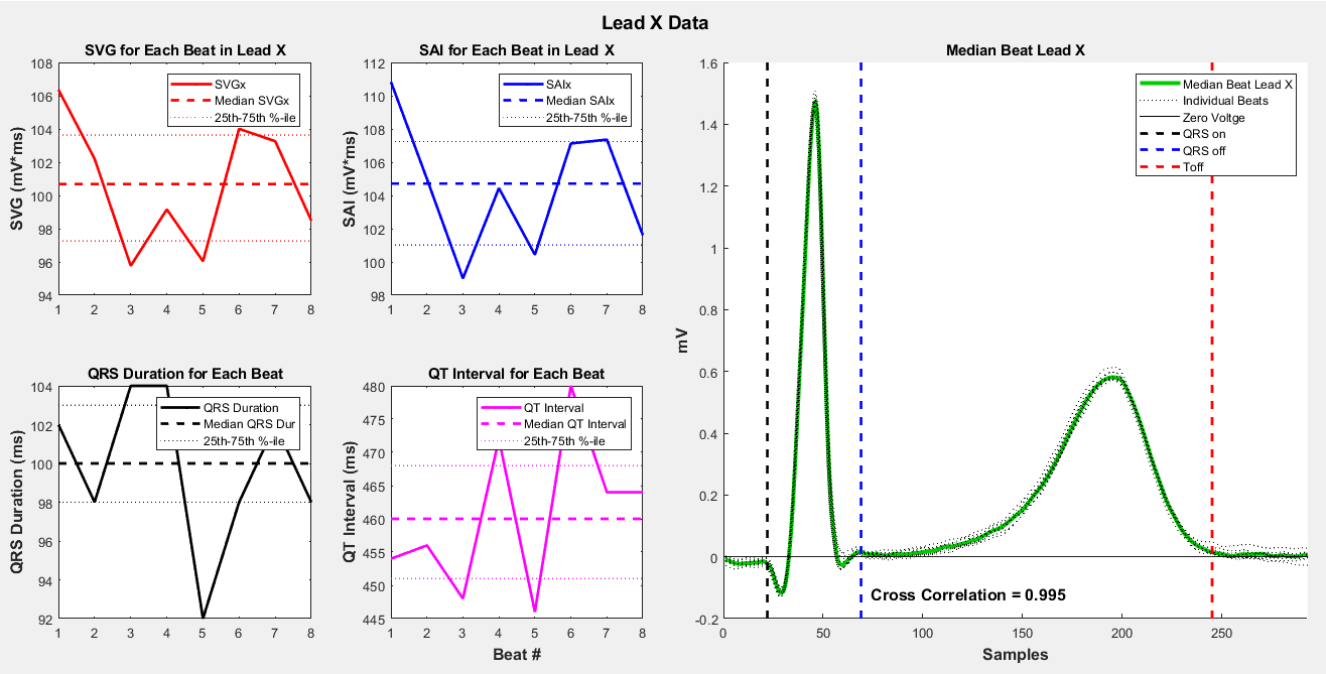


Figure 63: Individual beat stats for lead X in `example1.xml`. The QRS duration, QT interval, area under the QRST complex (SVG), and absolute area under the QRST complex (SAI) are shown. The right panel shows the median beat in green and the individual beats the make up the median beat in black. The NCC (0.995) is displayed.

20.11 Full VCG

This displays the "rhythm strip" VCG rather than the median beat VCG.

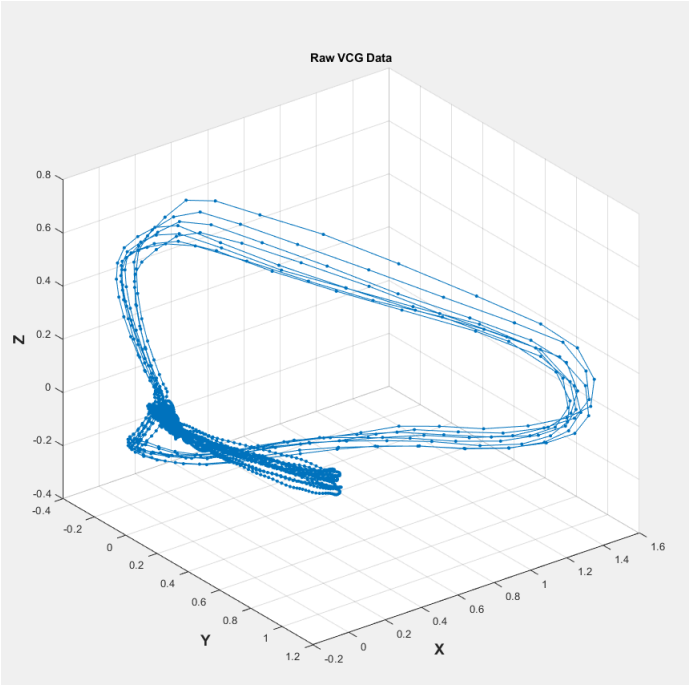


Figure 64: Raw VCG plotted in 3D for `example1.xml`.

20.12 Offsets

The Offsets button shows the effect of baseline correction, which is a process that attempts to set the TP segment, which is a true isoelectric interval of the ECG, to 0 voltage. An example of the offsets figure is shown in **Figure 40**. Further details can be found in **Chapter 15.3**.

20.13 Filtering

The Filtering button displays how the selected filtering parameters affects the loaded ECG. The original, unfiltered ECG signals are shown in black, and the signal post filtering (and baseline correction) is shown in red. The approximation of baseline wander is shown as a dashed blue line. An example filtering figure is shown here:

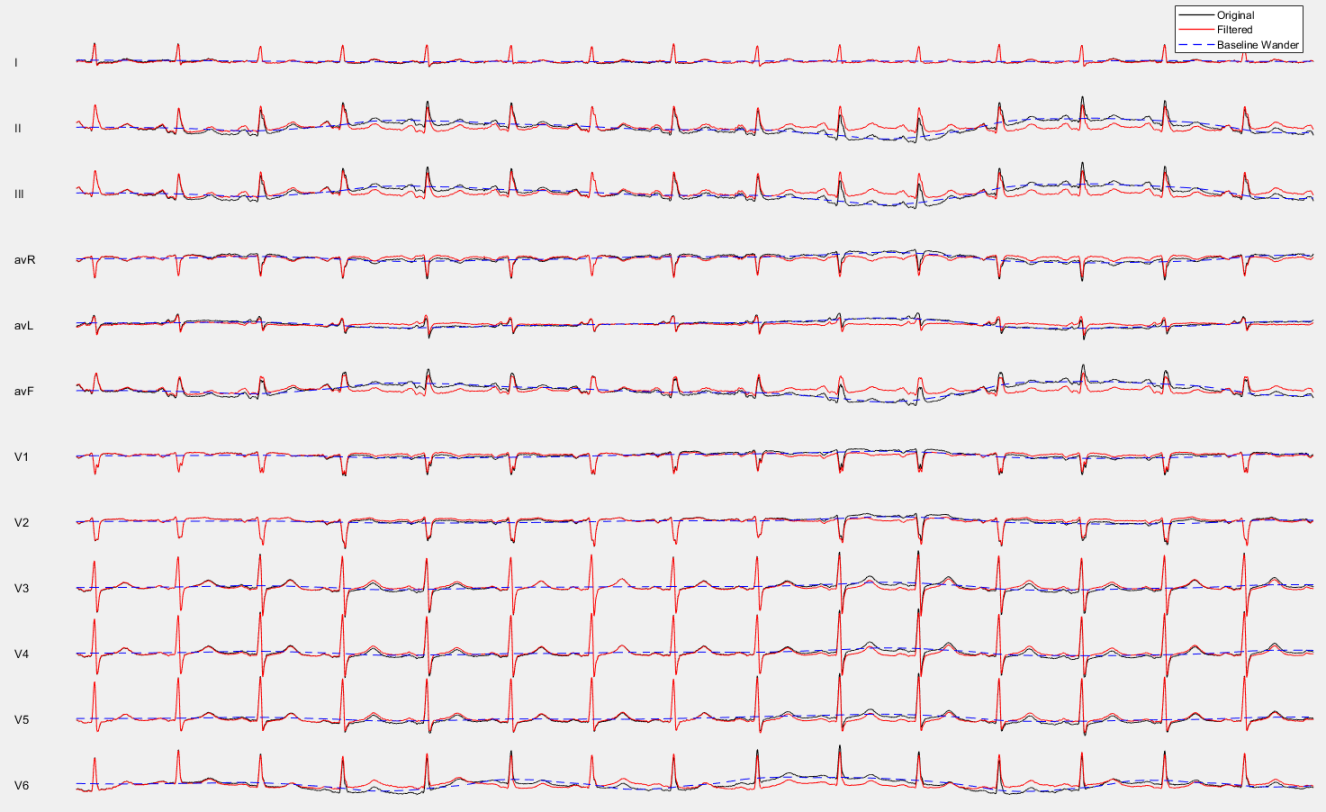


Figure 65: Filtering figure. The original, non-filtered, raw signal is shown in black, and the filtered signal is shown in red. The dashed blue line is the estimated baseline wander. In this example, note how leads II, III, avF, and V6 have significant baseline wander that is successfully removed with high-pass filtering. See filtering sub-figure (**Figure 66**) for additional detail.

In addition to the filtering figure shown above, clicking on the Filtering button beings up a sub-figure that displays the estimated signal to noise ratio (SNR) to estimate high frequency noise, and an estimation of low frequency noise/wander. *High* values of SNR are indicative of minimal high frequency noise, and *low* values of baseline wander estimates indicate minimal baseline wander. The cutoffs that are used to indicate good quality (green boxes) vs poor quality (red boxes) are set with quality parameters (`hf_noise` and `lf_noise`) in the external file `quality_presets.csv`. See **Chapter 16** for additional information.

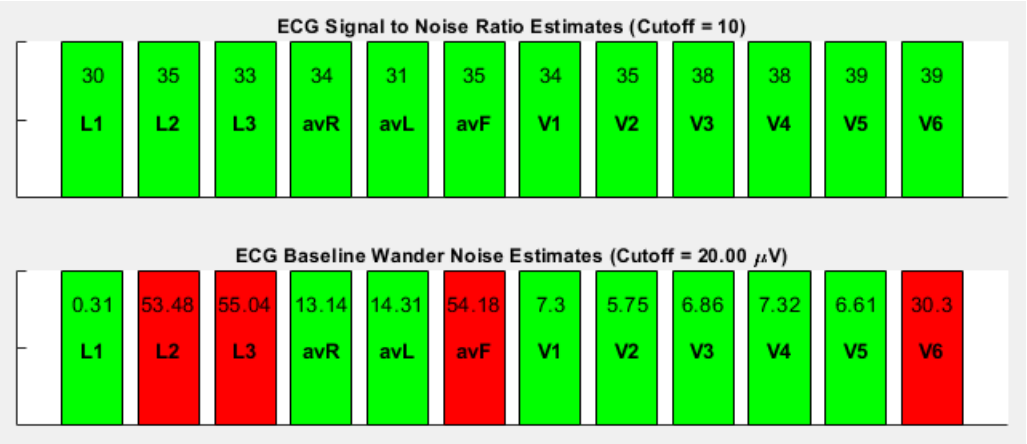


Figure 66: Filtering sub-figure for **Figure 65**. Note how in **Figure 65** leads II, III, avF, and V6 have significant baseline wander, and the sub-figure shows these leads have low-frequency noise above the set cutoff of 20 μV (red bars).

20.14 Threshold

The Threshold button displays the R peak threshold figure which is described in detail in **Chapter 15.1**, **Figure 24**, and **Figure 25**.

20.15 Pacing Spike Removal and Interpolation

If you are utilizing the CWT Filter for pacemaker spike removal (see **Chapter 5.4**, **Chapter 12.2**, and **Chapter 15.2**) the Show button in GUI section 1.18 can be used to visualize pacemaker spike detection and interpolation (if ☑ **Interpolation** is checked). See **Chapter 15.2** for additional information. An example using `example4.xml` is shown here:

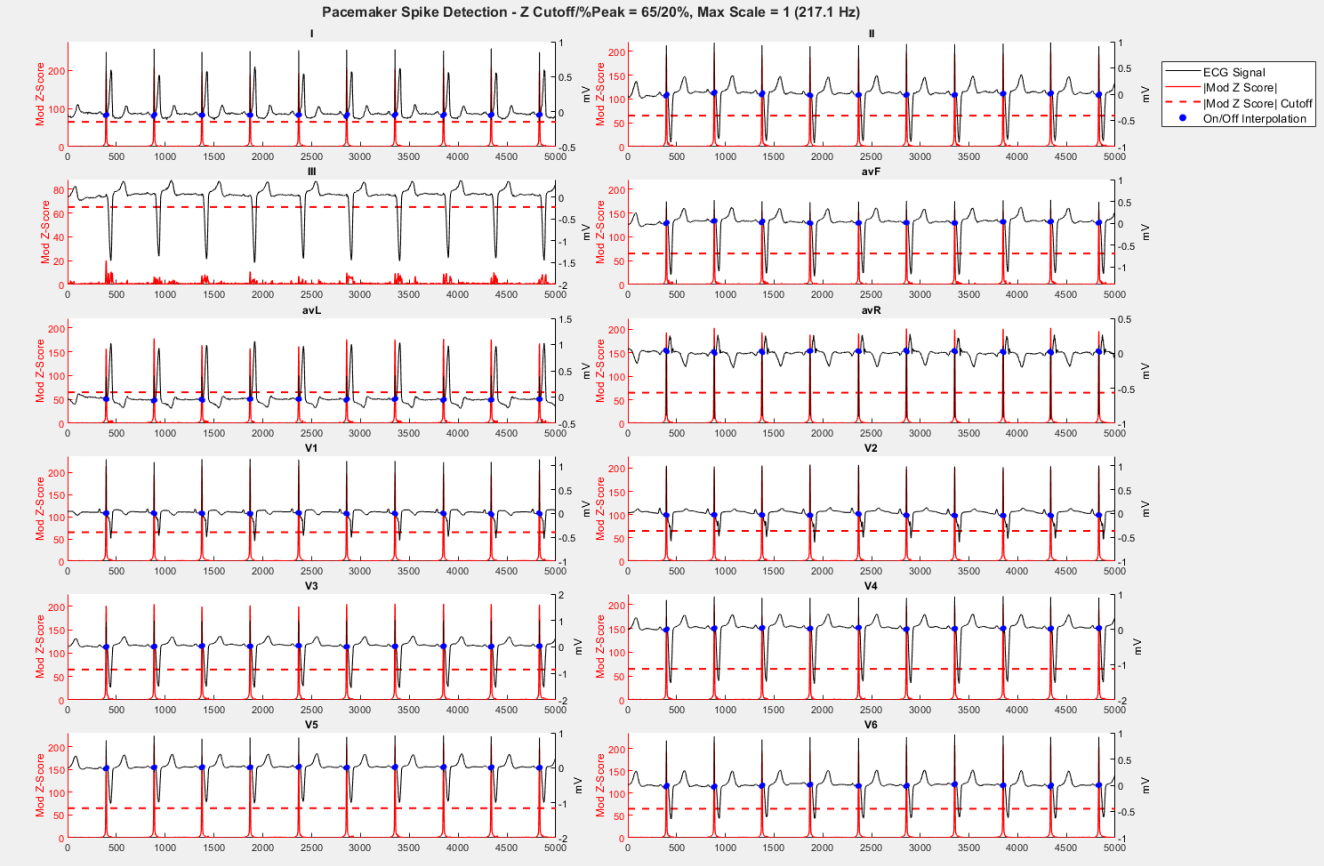


Figure 67: The Pacemaker Spike Detection figure for `example4.xml`. The absolute value of modified Z scores for the 1st difference of the ECG leads as described in **Chapter 15.2** is shown in red. Any time the modified Z score crosses the red horizontal dashed line (which is the value of `pacer_zcut` in `Annoparams.m`) a pacing spike is counted The onset and offset of the spike, as determined by `pacer_zpk` in `Annoparams.m`, a blue dot is marked on the ECG. The values for `pacer_zcut`, `pacer_zpk`, and `pacer_maxscale` from `Annoparams.m` are shown in the title. The frequency noted is the lower limit of the pseudo-frequency cutoff set by `pacer_maxscale`, such that all frequencies below this value are excluded from absolute modified Z score calculation.

As noted above, if **☑ Interpolation** is checked a second figure will also appear showing how each ECG lead's pacing spikes were removed with interpolation (see **Chapter 15.2** for further details):

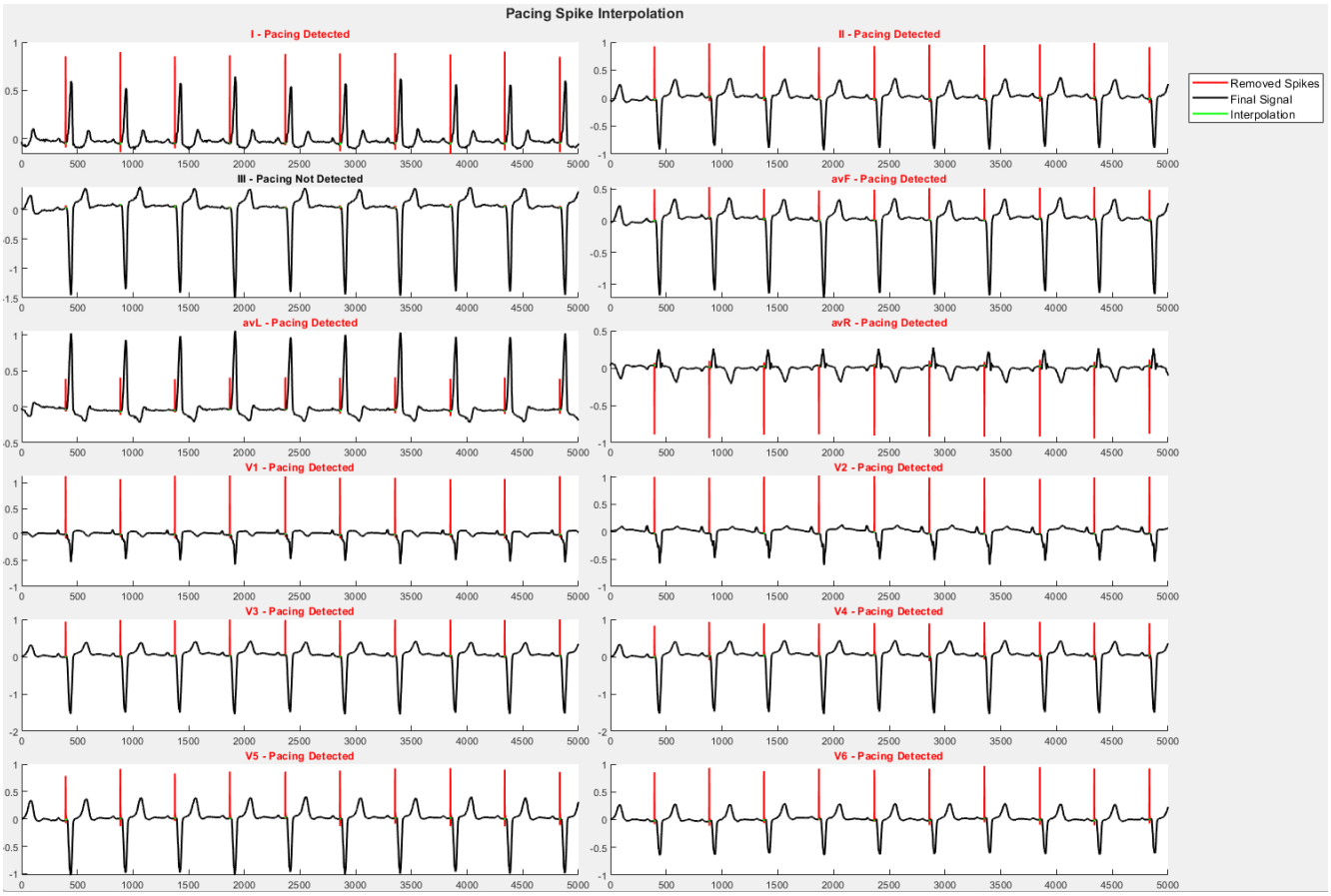


Figure 68: The Pacing Spike Interpolation figure for `example4.xml`. The original ECG signal in each lead is shown in black and the pacing spikes that are removed are shown in red. The final signal that is the result of interpolation is shown in green (may have to zoom in to see it). Text above each lead notes if pacing spikes were detected in that specific lead, but if spikes are detected in at least `pacer_spike_num` leads, any detected spike is removed in ALL leads regardless of if an individual spike was detected in every lead.

20.16 Debug Figures

The ☑ Debug checkbox enables debug figures that show information on first pass and median beat annotation that can help troubleshoot ECGs that are not processing to see if specific annotation parameters need to be adjusted. When ☑ Debug is checked, clicking **Calculate** brings up a figure showing the annotation markings/search windows for the full VM lead:

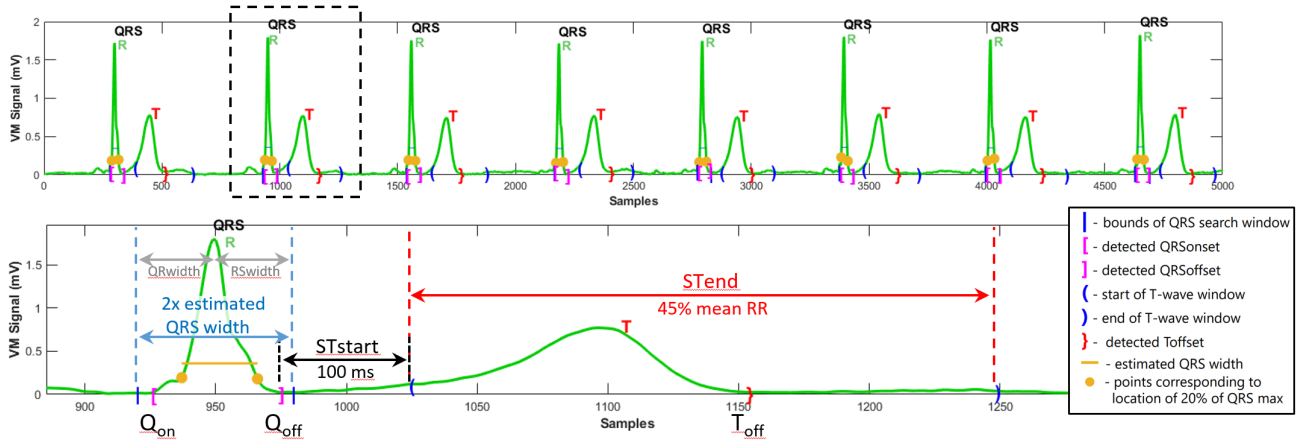


Figure 69: First pass annotation parameters and search windows. See **Chapter 18** for details.

As the median beat is annotated by a NN, the debug figure for median beat annotation shows the scores for each segment of the QRST complex being a "QRS complex", "T wave", or "other":

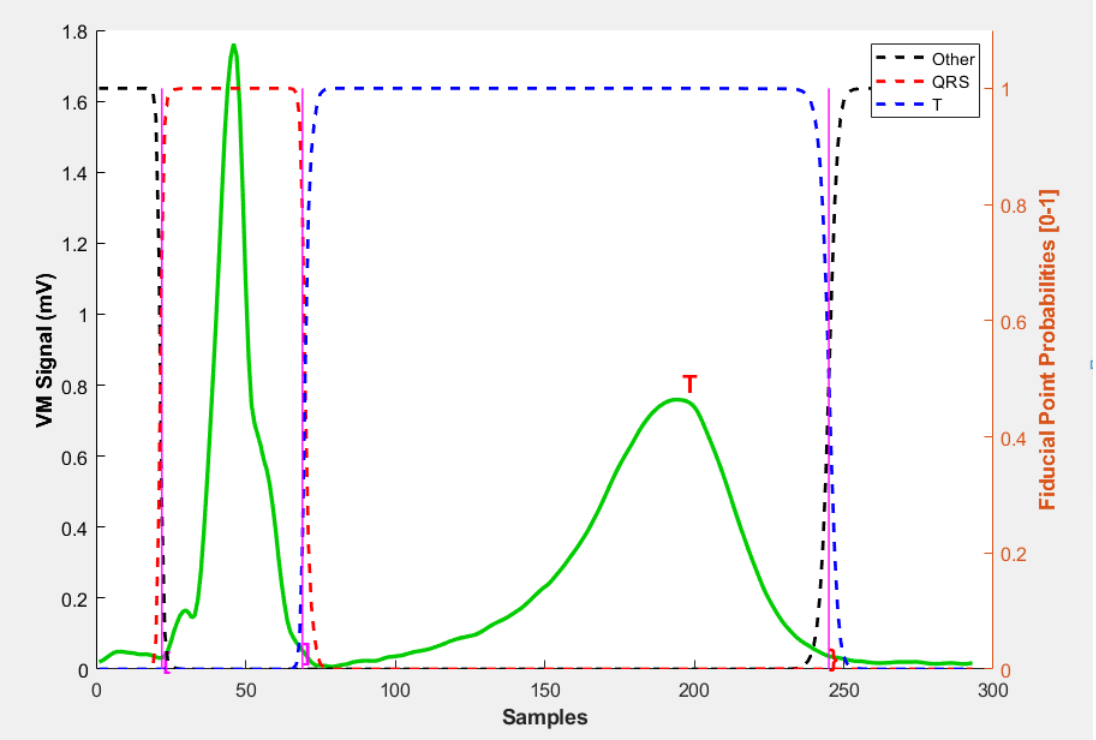


Figure 70: Debug figure for median beat annotation. The dashed lines are the NN predicted probabilities of a sample being a "QRS complex" (red), "T wave" (blue), or "other" (black). Each sample in the median beat signal is assigned the label that is the highest of the 3 probabilities. Q_{on} is set as the sample where "other" transitions to "QRS complex", Q_{off} is set as the sample where "QRS complex" transitions to "T wave", and T_{off} is set as the sample where "T wave" transitions to "other". Q_{on} is marked with a [, Q_{off} is marked with a], and T_{off} is marked with a } similarly to first pass debug figures. In rare cases where multiple potential fiducial points are detected, logic is used to select the best point – see methods manuscript [1] for additional details.

20.17 Angle Conventions

The 3D angle conventions are shown visually in **Figure 71** below:

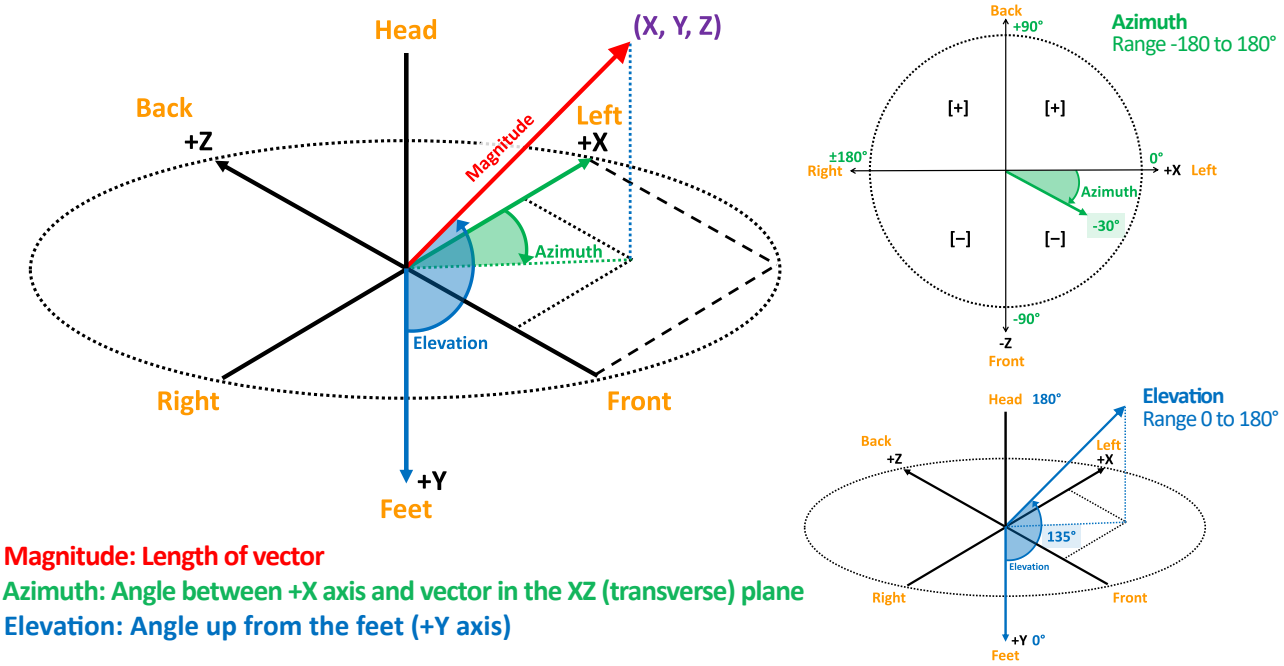


Figure 71: The following conventions for 3D angles are used by BRAVEHEART: Azimuth is the angle in the transverse (XZ) plane that can be between 0 and ±180°. Positive azimuth angles are posterior, and negative azimuth angles are anterior. Elevation is the angle up from the feet that can be between 0 and 180°. 0° points towards the feet and 180° points towards the head. Magnitude is the length of the vector.

This figure can be displayed in the GUI by clicking on the **Angle Info** button in GUI item **6.6.6** (see **Chapter 12.7**).

Slight modification of the standard definitions of azimuth (ϕ) and elevation (θ) are used by BRAVEHEART given that +X points leftward, +Y points downward, and +Z points posterior. To convert between spherical (r, θ, ϕ) and Cartesian (x, y, z) coordinates:

Cartesian to Spherical:	**Spherical to Cartesian:**
$r = \sqrt{x^2 + y^2 + z^2}$	$x = r\sin(\theta)\cos(\phi)$
$\theta = \arccos\left(\frac{y}{r}\right)$	$y = r\cos(\theta)$
$\phi = \arctan 2(z, x)$	$z = r\sin(\theta)\sin(\phi)$

21 Annotation Parameter Presets

Presets for first pass fiducial point detection can be saved to allow rapid switching between multiple parameters if needed. For example, if you are looking at ECGs with long QT syndrome, the nominal value of `STend` which is '45%' may result in missing the true ends of T waves, or if you want to look looking at tightly coupled PVCs, you may need to increase `maxBPM` from its nominal value of '150 bpm' and decrease `pkthresh` from its nominal value of '95'. You could manually adjust the value of `STend`, `maxBPM`, and/or `pkthresh` in `Annoparams.m` or using the GUI, but using presets allows you to save certain groups of parameters for later use.

Presents can only be used with the GUI version of BRAVEHEART. An alternative to use of manual adjustment of annotation parameters or use of presets is to specify non-nominal annotation parameters in an `.anno` file which allows much finer control over ECG/VCG processing and annotation than can be set with a preset (see **Chapter 17**).

Presets are set via the external file `search_presets.csv`. The file contains 7 columns separated by commas. Column 1 is the preset name that shows up in the BRAVEHEART GUI, and the next 6 columns correspond to the values of `pkthresh`, `maxBPM`, `QRwidth`, `RSwidth`, `STstart`, and `STend`, in that order as shown here:

```
name,       pkthresh,  maxBPM,  QRwidth,  RSwidth,  STstart,  STend
Standard,   95,        150,     100,      100,      100,      45
PVCs,       85,        200,     150,      150,      150,      35
BBB/Paced,  95,        150,     150,      150,      150,      45
Bigeminy,   85,        170,     150,      150,      150,      70
```

To add a new preset, simply add a new line with a preset name and the appropriate values of `pkthresh`, `maxBPM`, `QRwidth`, `RSwidth`, `STstart`, and `STend` in the appropriate columns, save the file, and restart BRAVEHEART (presets are only loaded when the BRAVEHEART GUI initializes).

To use a preset, select the preset name from dropdown GUI item **3.5**:

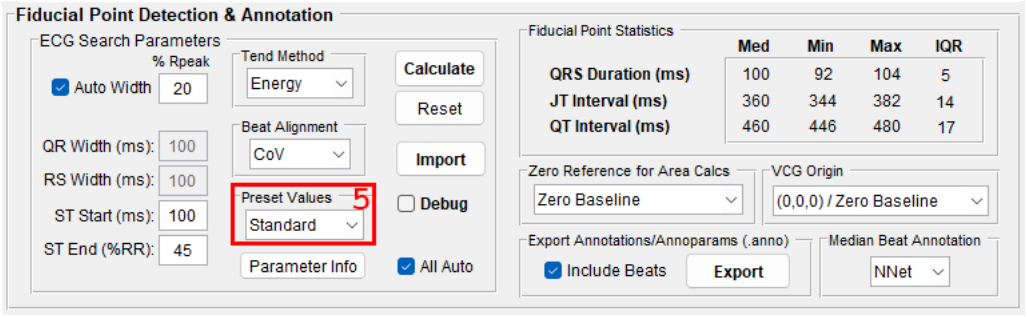


Figure 72: Preset dropdown in GUI

22 Normal Ranges

The BRAVEHART GUI can calculate normal ranges for a select number of VCG parameters based on age, sex, race, and BMI. Normal ranges were obtained by measuring VCG parameters on 3,292 patients with normal ECGs and no history of cardiovascular, pulmonary, or renal disease, and no abnormal cardiovascular imaging [2].

The following 10 parameters have normal ranges which can be displayed in the GUI:

- SVG area magnitude – `svg_area_mag` (see **Figure 71**)
- SVG area azimuth – `svg_area_az` (see **Figure 71**)
- SVG area elevation – `svg_area_el` (see **Figure 71**)
- SVG area X – `svg_x`
- SVG area Y – `svg_y`
- SVG area Z – `svg_z`
- SAI QRST – `sai_qrst`
- SAI VM – `sai_vm`
- Area (mean) QRST angle – `qrst_angle_area`
- Peak QRST angle – `qrst_angle_peak`

Further details of each parameter can be found in **Chapter 24**.

The BRAVEHEART GUI can extract age, sex, and race from ECG formats that contain this data, such as `.xml` or `.dcm`, using `xml_demographics.m`. See **Chapter 12.7** for additional details.

If you want to obtain normal ranges for use in a study, the Normal Values figure (see **Chapter 20.4** and **Figure 55**) can be used to display the normal ranges for a specific ECG loaded into the GUI. If you have an external file with age, sex, race, BMI, and heart rate, and if you have access to MATLAB, you can run the `NormalVals()` function to output a structure with the normal ranges for an input of age, sex, race, BMI, and heart rate.

The `NormalVals()` function can be used as follows:

```
vals = NormalVals(age, male, white, bmi, hr)
```

where `age` is the patient's age in years, `male = 1` if the patient is male and `male = 0` if the patient is not male, `bmi` is the body mass index, `hr` is the heart rate on the ECG, and `white = 1` if the patient is white and `white = 0` if the patient is not white.

For example, to obtain the normal values of a 50 year old white female with BMI 30 and a heart rate of 65 bpm, you would call the function as:

```
vals = NormalVals(50, 0, 1, 30, 65)
```

and the output would be a structure called `vals`:

```
vals =

  NormalVals with properties:

                age: 50
               male: 1
                bmi: 30
              white: 1
                 hr: 65
       svg_area_mag: [32.8758 103.7804]
        svg_area_az: [-46.1098 12.5448]
        svg_area_el: [44.0672 79.6125]
             sai_vm: [57.9000 127.9246]
          sai_qrst: [88.9246 200.2173]
             svg_x: [25.0370 84.4236]
             svg_y: [7.3287 57.5451]
             svg_z: [-41.2834 6.8222]
    qrst_angle_area: [11.3821 103.8678]
    qrst_angle_peak: [4.9115 96.8858]
```

The structure `vals` contains fields which describe the input parameters (age, sex, race, BMI, and heart rate), as well as the lower and upper bounds for the 10 parameters for which normal ranges are available. The first number in brackets is the lower bound of the normal range, and the second number in brackets is the upper bound for the normal range. For further details please see our publication where these values were derived [2].

Normal Ranges with Missing Demographic Information

If you do not have complete information on some of the inputs for `NormalVals()` you should leave that input empty by using `[]`, in which case that parameter is marganalized in the regression. If a value is left blank when calling `NormalVals()` the values for the input in question that is displayed within the output `vals` will have a value assigned to it that is the mean value for the study population that was used to derive the normal range regression. For the binary variables `male` and `white` this can result in values that are between 0 and 1.

23 ECG Format Information

BRAVEHEART can read a variety of ECG file formats. This section includes brief descriptions of the file formats that can be read by BRAVEHEART without modification. Additional file formats can be added as described in **Chapter 8**. Some formats may have multiple versions that are formatted slightly differently and which may result in errors when loading ECGs. We recommend that you verify your ECG format loads properly before processing large batches of ECGs.

Unformatted .txt

Unformatted text files (extension .txt) use `format = 'unformatted'` and load via the file `load_ecg.m`. These files contain 12 columns with data in the order of lead I, II, III, avR, avL, avF, V1. V2, V3, V4, V5, and V6. The sampling frequency is encoded in the file as the first value in each column, and the ECG signal data starts in row 2 for each column. The file's units are mV, so no conversion is needed. This is a simple file format, but the downside is that file sizes tend to be larger than if the same signals were stored using encoded .xml or other formats.

BIDMC .txt

BIDMC formatted text files (extension .txt) use `format = 'bidmc_txt'` and load via the file `load_ecg.m`. This file format is used to export ECGs using the legacy ECG system at BIDMC. Files contain 13 columns with the first column being the sample number starting at 1, and the next 12 columns being the data in the order of lead I, II, III, avR, avL, avF, V1, V2, V3, V4, V5, and V6. Sampling frequency
is 500 Hz and is not encoded in the file. The values at each sample are in units where 1 unit = 5 μV.

Prucka .txt

General Electric (GE) Prucka recording system formatted text files (extension .txt) use `format = 'prucka_txt'` and load via the file `load_ecg.m`. This file format is generated when exporting ECGs from the GE MacLab/Prucka electrophysiology recording system. These files contain 12 columns with data in the order of lead I, II, III, avR, avL, avF, V1, V2, V3, V4, V5, and V6. Sampling
frequency is 997 Hz and is not encoded in the file. The file units are mV, so no conversion is needed.

MUSE .xml

GE MUSE XML files (extension .xml) use `format = 'muse_xml'` and load via the file `load_musexml.m`. Sampling frequency is usually 500 Hz and is encoded in the XML. Lead data are stored in the relevant XML sections in base64 encoding in units where 1 unit = 4.88 μV which is also encoded in the XML. Files that have not been deidentified or had identifiers removed include age, sex, and race encoded in the XML.

Philips .xml

Philips XML files (extension .xml) use `format = 'philips_xml'` and load via the file `load_philipsxml.m`. Sampling frequency is usually 500 Hz and is encoded in the XML. Lead data are stored in the relevant XML sections in base64 encoding followed by LZW compression, in units where 1 unit is usually 5 μV which is also encoded in the XML. Files that have not been deidentified or had identifiers removed include age and gender encoded in the XML. We have found multiple versions of the Philips XML format; some include 10 seconds of data, while others contain 11 seconds of data which includes a 10 second ECG signal followed by a 1 second calibration signal which is cut off during loading.

HL7 .xml

HL7 XML files (extension .xml) use `format = 'hl7_xml'` and load via the file `load_hl7xml.m`. Sampling frequency, units per mV, and lead data are all encoded within the XML file. Files that have not been deidentified or had identifiers removed include age, sex, and race encoded in the XML. Details of the HL7 format standards can be found at `https://www.amps-llc.com/uploads/2017-12-7/aECG_Implementation_Guide.pdf`.

DICOM .dcm

DICOM files (extension .dcm) use `format = 'DICOM'` and load via the file `load_dicom.m`. Sampling frequency, units per mV, and lead data are all encoded in the DICOM file. Details of the DICOM format standards can be found at `https://www.dicomstandard.org/News-dir/ftsup/docs/sups/sup30.pdf`.
Note: `load_dicom.m` requires the MATLAB Image Processing Toolbox. If this toolbox is not available you will need to write your own function to load DICOM files.

ISHNE .ecg

International Society for Holter and Noninvasive Electrocardiology (ISHNE) files (extension .ecg) use `format = 'ISHNE'` and load via the file `load_ISHNE.m`. Sampling rate, signal resolution, and lead data are all encoded in the file. Details of the format can be found at `http://thew-project.org/papers/Badilini.ISHNE.Holter.Standard.pdf`.

GE .mrq

GE Marquee ASCII files (extension .mrq) use `format = 'mrq_ascii'` and load via the file `load_mrq.m`. Sampling frequency is 500 Hz and is not encoded in the file. Units are in μV.

Unformatted .csv

Unformatted comma separated value files (extension .csv) use `format = 'generic_csv'` and load via the file `load_unformatted.m`. This file format is unique in that the sampling frequency, units per mV, and the orientation of the data (rows or columns and lead order) are read from an external file called `generic_csv_params.csv` rather than within `ECG12.m`. This allows users to customize the frequency and signal resolution when using the compiled executable if needed, and allows more files to be read without having to manually edit code. Prior to version 1.2.2 the only way to change the frequency and units per mV was to edit the code in `ECG12.m`.

To set the sampling frequency, units per mV, data orientation (columns or rows), data offsets, and lead order, edit `generic_csv_params.csv` which is stored in the main directory with the following default values:

```
freq,         500
unitspermv,   1
orientation,  cols
row_start,    1
col_start,    1
lead_order,   I II III AVR AVL AVF V1 V2 V3 V4 V5 V6
```

To change the frequency and units per mV (make sure you are using units per mV and not mV per unit!) edit the numeric values. The value for `orientation` can be `cols` or `rows` depending on if your data is stored in columns or rows. The values for `row_start` and `col_start` can be changed to specify where the actual numeric data starts if there are any headers or other rows/columns that are used for labeling the data. For example, if rows 1-3 contain header information and column 1 contains a time stamp, your would set `row_start = 4` and

`col_start = 2`. `lead_order` simply specifies the order of the leads (regardless of columns or rows) with a space between each lead name. Capitalization of lead names is not required. If only 8 leads (I,II, V1-V6) are included, BRAVEHEART will automatically calculate the remaining 4 limb leads (III, avR, avL, avF).

Cardiosoft .xml

General Electric (GE) Cardiosoft version 6 XML files (extension .xml) use `format = 'cardiosoft_xml'` and load via the file `load_cardiosoftxml.m`. Sampling frequency is usually 500 Hz and is encoded in the XML. Lead data are stored in the relevant XML sections in units where 1 unit = 5 μV which is also encoded in the XML. Files that have not been deidentified or had identifiers removed include age, sex, and race encoded in the XML.

Schiller .xml

Schiller XML files (extension .xml) use `format = 'schiller_xml'` and load via the file `load_schillerxml.m`. Sampling frequency is usually 500 Hz and is encoded in the XML. Lead data are stored in the relevant XML sections in units where 1 unit = 1 μV which is also encoded in the XML. Files that have not been deidentified or had identifiers removed include age, sex, and race encoded in the XML.

SCP-ECG .scp

SCP-ECG files are binary files with extension .scp that have the benefit of good data compression and small file size at the expense of a complex file structure. SCP-ECG files use `format = 'scp_ecg'` and load via the file `load_scpecg.m`. Sampling frequency, units per mV, and lead data are all encoded in the .scp file. To load SCP-ECG files, BRAVEHEART utilizes code from The BioSig Project v3.8.4 (`https://biosig.sourceforge.net`) under GNU GPL License v3.

Physionet .csv

Physionet (`https://physionet.org/`) comma separated value files (extension .csv) use `format = 'physionet_csv'` and load via the file `load_physionet_csv.m`. Sampling frequency, units per mV, and lead data are all encoded in the .csv file

---

Physionet .dat

Physionet (`https://physionet.org/`) data files (extension .dat) use `format = 'physionet_dat'` and load via the file `load_physionet_dat.m`. Use of these .dat files requires that there is a header file (extension .hea) with the same basename as the .dat file in the same directory. For example, `ecg123.dat` requires that the header file `ecg123.hea` are in the same directory. Sampling frequency, units per mV, and lead data are all encoded in the .hea file. As of now, the only .dat file formats/encodings that have been tested are "Format 16" (see `https://physionet.org/physiotools/wag/signal-5.htm` for details), although other formats are supported based on data within the .hea file.

European Data Format (EDF) .edf

EDF ECG files are binary files with extension .edf that use `format = 'edf'` and load via the file `load_edf.m`. Sampling frequency, units per mV, and lead data are all encoded in the .edf file. Of note, .edf files from Physionet will often include some amount of artifact appended on the end of each ECG lead signal. This artifact is always a single value that is usually equal to or a multiple of the minimum value of physical units or digital units within the file. The inclusion of these artifacts can be problematic, because they can be orders of magnitude larger than the ECG signals themselves, and this can cause issues loading and processing the ECG. Currently, the `load_edf.m` module will assess if such a constant signal that is at least 50 ms long and less than 1 .edf data record duration is appended on the end of the ECG, and it so, it will remove it. If you are using .edf files you should be aware that this constant signal removal could theoretically cause an issue if one of more leads are missing/disconnected at the end of the recording, so that the segment of missing data is short. The current iteration of `load_edf.m` will remove this missing segment from ALL leads, which may not be desirable. You can detect if this happens by noting the number of samples in the output file - if this is less than expected and you are using .edf files, this is a likely cause. Additionally, to avoid ECGs not loading at all if a single lead is completely missing, if the constant artifact is longer than one .edf data record duration, it will NOT be removed as the observed artifacts were never longer than this during testing.

Norav 1200M Raw Data .rdt

Raw data type files from the Norav 1200M ECG recorder (extension .rdt) use `format = 'norav'` and load via the file `load_norav.m`. Sampling frequency is 500 Hz and is not encoded in the file. The values at each sample are in units where 1 unit = 2.44 μV.

Megacare .xml

Dräger Megacare XML files (extension .xml) use `format = 'megacare_xml'` and load via the file `load_megacarexml.m`. Sampling frequency, lead data, and units per mV are encoded in the XML.

Edan SE 601C Raw Data .dat

Edan raw data files (extension .dat) use `format = 'edan_dat'` and load via the file `load_edandat.m`. Sampling frequency (1000 Hz) and and signal resolution (2.52 μV per unit) are is not encoded in the .dat files. These values can be edited in `load_edandat.m` if needed.

Medical Waveform Format Encoding Rules (MFER) .mwf

MFER files (extension .mwf) use `format = 'mfer'` and load via the file `load_mfer.m`. Sampling frequency, lead data, and units per mV are encoded in the files.

BRAVEHEART .mat

BRAVEHEART exported ECG/VCG data (extension .mat) can be reloaded by using `format = 'braveheart_mat'` and load via the file `load_braveheart_mat.m`. Sampling frequency is encoded in the file. Signals are always stored mV. Further details of saving data in BRAVEHEART format can be found in **Chapter 25**.

Claris .txt

Abbott Workmate Claris ASCII text files (extension .txt) use `format = 'claris_txt'` and load via the file `load_claris.m`. Claris exports each lead as a **separate** text file, and the `load_claris.m` function will find all 12 (or 8) individual leads exported from the same page regardless of which file/lead is chosen during loading. Files must be in the format: `<Lead>.Session <K> - Page <P>.<N>.TXT`, where `<Lead>` is one of the 12 leads, `<K>` is some number related to the Claris session that will be the same for all files, `<P>` is the page in Claris that contained the exported signals, and `<N>` is the order of the signals on the Claris page. For example an exported 12-lead ECG may contain the following files:

`I.Session 8 - Page 1.1.TXT`

`II.Session 8 - Page 1.2.TXT`

`...`

`V6.Session 8 - Page 1.12.TXT`

for a 12-lead ECG displayed on Claris page 1. The text files do not contain any information on

the identification of the lead, and the lead name for each file is pulled from the filename. If files are renamed to a different file name structure they will not load properly.

`load_claris.m` will also check to see if there is Session Information file (such as `Session 8 Information.TXT` within the same directory, and with the same session number, that contains information on sampling frequency and units per mV. If so, it will use the values contained within this file for sampling frequency and units per mV. If this files is not present, the default values of 2000 Hz and 78 nV/unit are used.

No matter which of the 12-leads for a given ECG are selected to load into BRAVEHEART, all other files will be found and used to load their respective leads. For example, the same 12-lead is loaded no mater if the user loads the file `I.Session 8 - Page 1.1.TXT` or `V6.Session 8 - Page 1.12.TXT`. For this reason, it is critical that all 12 files that make up the 12-lead ECG are in the same directory and that their file names are in the correct format (see above). If the 8 independent leads are available rather than a full 12-lead ECG, the missing limb leads will be calculated.

Note that because the Workmate Claris EP recording system exports each lead as an individual file, if you run batch (via the GUI or command line), the software will see each lead as a new ECG, and therefore each "ECG" will show up in the results 12 times (or 8 if you only supply leads I,II, and V1-V6). We are working on a solution to this - in the interim, the batch process will take longer than it should, and you will have to delete the duplicate rows for each ECG lead file. Additionally, because the Information file has a .txt extension and has to be in the same directory as the Claris ASCII files, BRAVEHEART batch will try to read this file as an ECG and it will throw an error that will be noted in the `errors_<timestamp>.csv`. Alternatively, you can convert the Claris ASCII format into some other format that utilizes a single file.

24 Result Class Variables

Variables and their definitions for the 3 results classes are presented here. Detailed mathematical explanations and associated equations are available in the Methods Manuscript.

24.1 Lead Morphology Class (`Lead_Morphology.m`)

[lead] refers to any of the 16 leads (L1, L2, L3, avR, avL, avF, V1-V6, X, Y, Z, VM)

Variable	Description	Units
`[lead]_r_wave`	Magnitude of R wave on median beat of `[lead]`	mV
`[lead]_s_wave`	Magnitude of S wave on median beat of `[lead]`	mV
`[lead]_rs_wave`	Magnitude of entire QRS complex = `[lead]_r_wave` + `abs([lead]_s_wave`	mV
`[lead]_rs_ratio`	Ratio of R wave to magnitude of entire QRS complex = `[lead]_r_wave` / `[lead]_rs_wave`	–
`[lead]_sr_ratio`	Ratio of S wave to magnitude of entire QRS complex = `[lead]_s_wave` / `[lead]_rs_wave`	–
`[lead]_t_max`	Maximum magnitude of T wave in `[lead]`	mV
`[lead]_t_max_loc`	Timing of T wave maximum (after QRS onset) in `[lead]`	ms
`[lead]_qrs_area`	Area of `[lead]` median beat QRS complex	mV·ms
`[lead]_t_area`	Area of `[lead]` median beat T wave	mV·ms
`[lead]_qrst_area`	Area of `[lead]` median beat full QRST complex	mV·ms
`VM_max_rpk_loc`	Timing of the maximum R wave in lead VM (maximum distance from origin)	ms
`cornell_lvh_mv`	Cornell LVH = `V3_s_wave` + `avL_r_wave`	mV
`sokolow_lvh_mv`	Sokolow-Lyon LVH = `V1_s_wave` + max(`V5_r_wave`, `V6_r_wave`)	mV
`qrs_frontal_axis`	Fontal plane QRS axis	deg

24.2 VCG Morphology Class (`VCG_Morphology.m`)

Variable	Description	Units
TCRT	Total Cosine R to T (range 0 - 1)	–
TCRT_angle	Angle from TCRT = acos(TCRT)	deg
qrsloop_residual	SVD variance from fitting QRS loop to a plane (0 = perfect fit) = qrs_S3^2	–
qrsloop_rmse	RMSE for fit of QRS loop to best fit plane (0 = perfect fit)	mV
qrsloop_roundness	QRS loop roundness. 1 = perfect circle, larger values are increasingly elliptical = qrs_S1/qrs_S2	–
qrsloop_area	Area of QRS loop	mV
qrsloop_perimeter	Length of QRS loop projected into best fit plane	mV^2
tloop_residual	SVD variance from fitting T loop to a plane (0 = perfect fit) = t_S3^2	–
tloop_rmse	RMSE for fit of T loop to best fit plane (0 = perfect fit)	mV
tloop_roundness	T loop roundness. 1 = perfect circle, larger values are increasingly elliptical = t_S1/t_S2	–
tloop_area	Area of T loop	mV
tloop_perimeter	Length of T loop projected into best fit plane	mV^2
qrs_loop_normal	Unit vector normal to best fit QRS loop plane	–
t_loop_normal	Unit vector normal to best fit T loop plane	–
qrst_dihedral_ang	Dihedral angle between best fit QRS loop and T loop planes	deg
qrs_loop_plane_az	Azimuth angle for unit vector normal to best fit QRS loop plane	deg
qrs_loop_plane_el	Elevation angle for unit vector normal to best fit QRS loop plane	deg
t_loop_plane_az	Azimuth angle for unit vector normal to best fit T loop plane	deg
t_loop_plane_az	Elevation angle for unit vector normal to best fit T loop plane	deg
qrs_S1	1st singular value of QRS loop	–
qrs_S2	2nd singular value of QRS loop	–
qrs_S3	3rd singular value of QRS loop	–
t_S1	1st singular value of T loop	–
t_S2	2nd singular value of T loop	–
t_S3	3rd singular value of T loop	–
qrs_var_s1_total	% of total variance made up by 1st QRS singular value	%
qrs_var_s2_total	% of total variance made up by 2nd QRS singular value	%
qrs_var_s3_total	% of total variance made up by 3rd QRS singular value	%
t_var_s1_total	% of total variance made up by 1st T singular value	%
t_var_s2_total	% of total variance made up by 2nd T singular value	%
t_var_s3_total	% of total variance made up by 3rd T singular value	%
TMD	T wave mechanical dispersion	deg
TWR_abs	Absolute T wave residuum	mV^2
TWR_rel	Relative T wave residuum	%
xy_qrs_loop_dir	Direction of rotation of QRS loop in XY plane ('CW', 'CCW', or 'Indeterminate')	–
xy_qrs_signed_area	Signed area of QRS loop in XY plane	mV^2
xz_qrs_loop_dir	Direction of rotation of QRS loop in XZ plane ('CW', 'CCW', or 'Indeterminate')	–
xz_qrs_signed_area	Signed area of QRS loop in XZ plane	mV^2
zy_qrs_loop_dir	Direction of rotation of QRS loop in ZY plane ('CW', 'CCW', or 'Indeterminate')	–
zy_qrs_signed_area	Signed area of QRS loop in ZY plane	mV^2
best_qrs_loop_dir	Direction of rotation of QRS loop in best fit plane ('CW', 'CCW', or 'Indeterminate')	–
best_qrs_signed_area	Signed area of QRS loop in best fit plane (absolute value should approximate qrsloop_area)	mV^2

24.3 VCG Calculation Class (`VCG_Calc.m`)

Variable	Description	Units
qrs_int	QRS duration	ms
qt_int	QT interval	ms
svg_x	X component of SVG = XQ_area + XT_area	mV·ms
svg_y	Y component of SVG = YQ_area + YT_area	mV·ms
svg_z	Z component of SVG = ZQ_area + ZT_area	mV·ms
sai_x	Area under absolute value of the median X QRST complex	mV·ms
sai_y	Area under absolute value of the median Y QRST complex	mV·ms
sai_z	Area under absolute value of the median Z QRST complex	mV·ms
sai_qrst	SAI QRST = sai_x + sai_y + sai_z	mV·ms
sai_vm	Area under median VM QRST complex (always positive)	mV·ms
q_peak_mag	Magnitude of peak QRS vector	mV
q_peak_az	Azimuth of peak QRS vector	deg
q_peak_el	Elevation of peak QRS vector	deg
t_peak_mag	Magnitude of peak T wave vector	mV
t_peak_az	Azimuth of peak T wave vector	deg
t_peak_el	Elevation of peak T wave vector	deg
svg_peak_mag	Magnitude of the sum of peak QRS and peak T vectors ("peak SVG")	mV
svg_peak_az	Azimuth of the "peak SVG" vector	deg
svg_peak_el	Elevation of the "peak SVG" vector	deg
q_area_mag	Magnitude of QRS area vector ([XQ_area, YQ_area, ZQ_area])	mV·ms
q_area_az	Azimuth of QRS area vector	deg
q_area_el	Elevation of QRS area vector	deg
t_area_mag	Magnitude of T-wave area vector [XT_area, YT_area, ZT_area]	mV·ms
t_area_az	Azimuth of T-wave area vector	deg
t_area_el	Elevation of T-wave area vector	deg
svg_area_mag	Magnitude of the SVG vector [svg_x, svg_z, svg_z]	mV·ms
svg_area_az	Azimuth of the SVG vector	deg
svg_area_el	Elevation of the SVG vector	deg
qrst_angle_area	Area (mean) QRST angle: 3D angle between area QRS and area T wave vectors	deg
qrst_angle_peak	Peak QRST angle: 3D angle between peak QRS and peak T wave vectors	deg
qrst_angle_peak_frontal	Projection of area QRST angle into frontal plane	deg
qrst_angle_area_frontal	Projection of peak QRST angle into frontal plane	deg
XQ_area	Area under median X QRS complex	mV·ms
YQ_area	Area under median Y QRS complex	mV·ms
ZQ_area	Area under median Z QRS complex	mV·ms
XT_area	Area under median X T wave	mV·ms
YT_area	Area under median Y T wave	mV·ms
ZT_area	Area under median Z T wave	mV·ms
VMQ_area	Area under median VM QRS complex	mV·ms
VMT_area	Area under median VM T wave	mV·ms

Variable	Description	Units
XQ_peak	Value of median X QRS complex at time of maximum distance from origin	mV
YQ_peak	Value of median Y QRS complex at time of maximum distance from origin	mV
ZQ_peak	Value of median Z QRS complex at time of maximum distance from origin	mV
XT_peak	Value of median X T wave at time of maximum distance from origin	mV
YT_peak	Value of median Y T wave at time of maximum distance from origin	mV
ZT_peak	Value of median Z T wave at time of maximum distance from origin	mV
speed_max	Maximum speed across the entire VCG loop	mV/ms
speed_min	Minimum speed across the entire VCG loop	mV/ms
speed_med	Median speed across the entire VCG loop	mV/ms
time_speed_max	Time after QRS onset of maximum VCG speed	ms
time_speed_min	Time after QRS onset of minimum VCG speed	ms
speed_qrs_max	Maximum speed across the QRS VCG loop	mV/ms
speed_qrs_min	Minimum speed across the QRS VCG loop	mV/ms
speed_qrs_med	Median speed across the QRS VCG loop	mV/ms
time_speed_qrs_max	Time after QRS onset of maximum QRS speed	ms
time_speed_qrs_min	Time after QRS onset of minimum QRS speed	ms
speed_t_max	Maximum speed across the T wave loop	mV/ms
speed_t_min	Minimum speed across the T wave loop	mV/ms
speed_t_med	Median speed across the T wave loop	mV/ms
time_speed_t_max	Time after QRS onset of maximum T-wave speed	ms
time_speed_t_min	Time after QRS onset of minimum T-wave speed	ms
sti_qrst	Speed-time integral for QRST complex	mV
sti_qrs	Speed-time integral for QRS complex	mV
sti_t	Speed-time integral for T Wave	mV
qrst_distance_area	Distance between the area QRS and area T vectors	mV
qrst_distance_peak	Distance between the area QRS and area T vectors	mV
vcg_length_qrs	Length of QRS VCG loop	mV
vcg_length_t	Length of T wave VCG loop	mV
vcg_length_qrst	Length of QRST VCG loop = vcg_length_qrs + vcg_length_t	mV
vm_tpeak_time	Time after QRS onset of peak of median VM Twave	ms
vm_tpeak_tend_abs_diff	Time difference between T wave peak and T wave end in median VM lead	ms
vm_tpeak_tend_ratio	Ratio between time of T wave peak and time of T wave end in median VM lead	–

24.4 Other Variables

Variable	Description	Units/Format
filename	ECG filename	–
version	BRAVEHEART version	–
date	Date single ECG data or batch data exported	MM/DD/YYYY
time	Date single ECG data or batch data exported	HH:mm
source	ECG file format string	–
freq	ECG sampling frequency	Hz
num_samples	number of samples in ECG file	–
num_beats	number of QRST complexes processed to make median beat	–
initial_num_beats	number of QRST complexes initially detected	–
num_pvcs_removed	number of beats removed due to PVC detection	–
num_outliers_removed	number of beats removed due to outlier detection	–
num_bad_removed	number of beats removed due to being incomplete (at very start/end of tracing)	–
hr	heart rate	bpm
cross_corr	Minimum normalized cross correlation for X, Y, and Z leads	–
noise_hf	Estimation of high-frequency noise in raw signal (minimum estimated SNR for all leads)	dB
noise_lf	Estimation of low-frequency noise/wander in raw signal by measuring variance of wander	mV
qual_prob	Probability of ECG being "good" quality (range 0-1)	–
missing_lead	Binary variable for a lead missing from the ECG	–
Q	Location in samples of $Q_{on}$for individual beats included in the median	Samples
QRS	Location in samples of R peaks for individual beats included in the median	Samples
S	Location in samples of $Q_{off}$for individual beats included in the median	Samples
Tend	Location in samples of $T_{off}$for individual beats included in the median	Samples
QRS_rem_pvc	Location in samples of R peaks for beats removed by PVC detection/removal	Samples
QRS_rem_outlier	Location in samples of R peaks for beats removed by outlier detection/removal	Samples
QRS_rem_manual	Location in samples of R peaks for beats removed manually	Samples
QRS_rem_bad	Location in samples of R peaks for beats removed due to being incomplete or overlapping	Samples
qrs_median	Median value for QRS duration for all individual beats included in the median	Samples
qrs_min	Minimum value for QRS duration for all individual beats included in the median	Samples
qrs_max	Maximum value for QRS duration for all individual beats included in the median	Samples
qrs_iqr	Interquartile range for QRS duration for all individual beats included in the median	Samples
jt_median	Median value for JT interval for all individual beats included in the median	Samples
jt_min	Minimum value for JT interval for all individual beats included in the median	Samples
jt_max	Maximum value for JT interval for all individual beats included in the median	Samples
jt_iqr	Interquartile range for JT interval for all individual beats included in the median	Samples
qt_median	Median value for QT interval for all individual beats included in the median	Samples
qt_min	Minimum value for QT interval for all individual beats included in the median	Samples
qt_max	Maximum value for QT interval for all individual beats included in the median	Samples
qt_iqr	Interquartile range for QT interval for all individual beats included in the median	Samples
RR_n	RR intervals only for beats included in median beat	ms
RR_pct_n	Percent change in subsequent RR intervals only for beats included in median beat	%
RR_sd_n	Successive differences in RR intervals only for beats included in median beat	ms
RR	RR intervals for all R peaks regardless of if they were removed	ms
RR_pct	Percent change in subsequent RR intervals regardless of if they were removed	%
RR_sd	Successive differences in RR intervals regardless of if they were removed	ms
SDNN	Standard deviation of RR intervals only for beats included in median beat	ms
SDRR	Standard deviation of RR intervals regardless of if they were removed	ms
RMSSD_n	Root mean square of successive differences only for beats included in median beat	ms
RMSSD_all	Root mean square of successive differences regardless of if they were removed	ms
pacing_detected	Flag for if pacing was detected (1) or not (0) using either pacemaker detection method	–
num_paced_leads	Number of leads in which pacing was detected if using CWT spike removal method	–

25 Signals/Data Export Format

BRAVEHEART will export a `.mat` file containing signal data (ECG/VCG signals, median beat signals, individual beats), and results by either clicking the Export Signals (.mat) button (GUI item 6.2.6), or during batch export if `save_data = 1` from the command line or ☑ Data is checked in the GUI (GUI item 1.18).

IMPORTANT: It is currently not possible to export these data in a format other than `.mat`. We have not been able to get the `.mat` files to load in current versions of Octave due to how Octave handles class objects differently than MATLAB.

Regardless of the name of the `.mat` file, when it is loaded into MATLAB, it will create a structure called `data` which has the following fields which are accessed using dot notation:

```
data =

  struct with fields:

            filename: 'C:\BRAVEHEART\Example ECGs\example1.xml'
          annoparams: [1x1 Annoparams]
             ecg_raw: [1x1 ECG12]
        ecg_filtered: [1x1 ECG12]
             vcg_raw: [1x1 VCG]
        vcg_filtered: [1x1 VCG]
  ecg_raw_postinterp: [1x1 ECG12]
        pacer_spikes: []
        lead_ispaced: [1x1 struct]
               beats: [1x1 Beats]
          beat_stats: [1x1 Beat_Stats]
                 geh: [1x1 VCG_Calc]
          median_vcg: [1x1 VCG]
          median_12L: [1x1 ECG12]
          medianbeat: [1x1 Beats]
    beats_median_vcg: [1x1 VCG]
    beats_median_12L: [1x1 ECG12]
        lead_morph: [1x1 Lead_Morphology]
         vcg_morph: [1x1 VCG_Morphology]
           quality: [1x1 Quality]
```

For example, to access the raw ecg signal ECG12 object, you would call

```
data.ecg_raw

ans =

  ECG12 with properties:

      hz: 500
   units: 'mV'
       I: [5000x1 double]
      II: [5000x1 double]
     III: [5000x1 double]
     avF: [5000x1 double]
     avL: [5000x1 double]
     avR: [5000x1 double]
      V1: [5000x1 double]
      V2: [5000x1 double]
      V3: [5000x1 double]
      V4: [5000x1 double]
      V5: [5000x1 double]
      V6: [5000x1 double]
```

and `data.ecg_raw.I` would return lead I from the raw ECG signal:

```
data.ecg_raw.I

ans =

   -0.0976
   -0.0976
   -0.0976
   -0.0976
   -0.0878
   -0.0781
   -0.0683
   -0.0586
   -0.0683
   -0.0781

   ...
```

Descriptions of the contents of the `data` structure are listed here in detail:

`filename`

The path and filename for the ECG that was processed.

`annoparams`

The annotation parmeters (Annoparams) used for ECG/VCG processing. This is an `Annoparams` object, which has the properties as outlined in **Chapter 6**.

`ecg_raw`

The original signals from the 12-lead ECG prior to any denoising/processing. This is an `ECG12` object, which has the properties as outlined in **Chapter 8.1**. These include the 12 ECG leads (`I` , `II` , `III` , `avR` , `avL` , `avF` , `V1` , `V2` , `V3` , `V4` , `V5` , `V6`), ECG sampling frequency (`hz`), and an optional text variable for signal units (`units`).

`ecg_filtered`

The denoised/baseline adjusted 12-lead ECG. This `ECG12` object has the same structure as `ecg_raw` .

`vcg_raw`

The VCG created from `ecg_raw` without any denoising/processing. This is a `VCG` object, which has the properties as outlined in **Chapter 10.1.1**. These include the VCG leads `X` , `Y` , `Z` , and `VM` , VCG sampling frequency (`hz`), and an optional text variable for signal units (`units`).

`vcg_filtered`

The denoised/baseline adjusted VCG. This `VCG` object has the same structure as `vcg_raw` .

`ecg_raw_postinterp`

The raw ECG signal (stored as an `ECG12` object) after pacemaker spike removal and interpolation has been performed if signal interpolation is turned on via `interpolation = 1` in `Annoparams.m` or by checking ☑ **Interpolation** in GUI item 1.18. If CWT spike removal is not active (see **Chapter 15.2**, this is the same data as in `ecg_raw` (see above). If pacemaker spike removal and interpolation is active, this is the signal

without pacemaker spikes that is subsequently filtered and used for analysis.

`pacer_spikes`

The raw pacing spikes which were detected by the CWT spike removal method as an `ECG12` object. Each signal is the length of the ECG and is mostly `NaN` with the detected pacemaker spikes as numeric values at their corresponding time points. If CWT spike removal is not active, this will appear empty as `[]`.

`lead_ispaced`

A structure with each of the 12 leads denoting if pacing in that lead was detected. When CWT spike removal is active (see **Chapter 15.2**, a value of 1 or 0 indicates if pacing was detected in that lead. If pacemaker spike detection via Width Filtering is active instead, if a spike is detected, values of -1 are assigned to ALL leads.

`beats`

The fiducial points for the individual QRST complexes that were used to make the median beat. This is a `beats` object that has the properties outlined in **Chapter 10.1.1**. The samples recorded within the fiducial point properties were used on the signals contained within `ecg_filtered` and `vcg_filtered` to align and create the median beats. The individual beats used to create the median beat are more easily accessible within the `beats_median_vcg` and `beats_median_vcg` structures within `data` as shown below.

`beats_stats`

The annotation parmeters (Annoparams) used for ECG/VCG processing. This is an `Annoparams` object, which has the properties as outlined in **Chapter 6**.

`geh`

Data from the `VCG_Calc` results class (see **Chapter 24.3**). Return a specific variable within the `VCG_Calc` results class with `data.geh.<variable>`. For example, the value of `svg_x` would be accessed as `data.geh.svg_x`.

`median_vcg`

The X, Y, X, and VM median beats which are used for all BRAVEHEART calculations. This `VCG` object has the same structure as `vcg_raw`, but only contains a single median

beat. Note that the median beats contain signal prior to QRS onset and after the T wave end; the fiducial points within `medianbeat` allow you to segment out the QRST complex.

`median_12L`

The 12-lead ECG median beats used for all BRAVEHEART calculations (primarily `Lead_Morphology`). This `ECG12` object has the same structure as `ecg_raw`, but only contains a single median beat. Note that the median beats contain signal prior to QRS onset and after the T wave end; the fiducial points within `medianbeat` allow you to segment out the QRST complex.

`medianbeat`

Fiducial points for the median beats in `median_vcg` and `median_12L`. The median beat signals contained in `median_vcg` and `median_12L` contain signal prior to QRS onset and after the T wave end, and the fiducial points within `medianbeat` allow you to segment out the QRS or QRST complex. Properties include:

- `Q` – QRS onset
- `QRS` – R peak (VM lead only) - for other leads use data in `Lead_Morphology`.
- `S` – QRS offset
- `T` – T peak (VM lead only) - for other leads use data in `Lead_Morphology`.
- `Tend` – T wave end

For example, to segment out the QRST complex from the VM lead you would code:

```
vm = data.median_vcg.VM;
qon = data.medianbeat.Q;
toff = data.medianbeat.Tend;
vm_qrst = vm(qon:toff);
```

`beats_median_vcg`

Aligned X, Y, Z, and VM beats which were used to calculated the median VCG beats in `median_vcg`. This is the same data as if you segmented out each beat using `vcg_filtered`. This `VCG` object has the same structure as `median_vcg`, but instead of `X`, `Y`, `Z`, and `VM` having a single beat, they have n beats where each beat is a different row within the property. For example, to access the 2nd beat that was used to make the lead X median beat, you would call `data.beats_median_vcg.X(2,:)`.

`beats_median_12L`

Aligned 12-lead beats which were used to calculated the median 12-lead ECG beats in `median_12L`. This is the same data as if you aligned and segmented out each beat using `ecg_filtered`. This `VCG` object has the same structure as `median_12L`, but instead of each lead having a single beat, they have n beats where each beat is a different row within the property. For example, to access the 3rd beat that was used to make the lead II median beat, you would call `data.beats_median_12L.II(3,:)`.

`lead_morph`

Data from the `Lead_Morphology` results class (see **Chapter 24.1**). Return a specific variable within the `Lead_Morphology` results class with `data.lead_morph.<variable>`. For example, the value of `L1_r_wave` would be accessed as `data.lead_morph.L1_r_wave`.

`vcg_morph`

Data from the `VCG_Morphology` results class (see **Chapter 24.2**). Return a specific variable within the `VCG_Morphology` results class with `data.vcg_morph.<variable>`. For example, the value of `TCRT` would be accessed as `data.vcg_morph.TCRT`.

`quality`

Data from ECG/VCG quality assessments. This is a `Quality` object which has the properties as outlined in **Chapter 16.3**. Note that the values contained in `data.quality` are the flags for if a specific quality parameter was out of range (value of 0 or 1), with the exception of `prob_value` which is the actual value of `quality_prob` with a numeric range of 0–1 (see **Chapter 16.1**).

---

26 Keyboard Shortcuts

When using the BRAVEHEART GUI the following keyboard Shortcuts can be used:

- F1 – Opens userguide (this document).
- Ctrl + l (lowercase L) – Equivalent to clicking **Load**.
- Ctrl + r – Equivalent to clicking **Reload**.
- Ctrl + e – Equivalent to clicking **Calculate**.
- Ctrl + s – Equivalent to clicking **Export Data**.
- Ctrl + b – Equivalent to clicking **Batch**
- Ctrl + d – Equivalent to clicking **Remove Beat**.
- Ctrl + a – Toggles the ☑ All Auto checkbox.
- Ctrl + ← – Equivalent to clicking ← to cycle through ECGs within a folder.
- Ctrl + → – Equivalent to clicking → to cycle through ECGs within a folder.
- Ctrl + ↑ – Equivalent to clicking Prev to cycle through beats.
- Ctrl + ↓ – Equivalent to clicking Next to cycle through beats.

Note that some Mac computers have a setting enabled called Mission Control that uses Ctrl + the arrow keys to navigate between different desktops and open windows. If this setting is enabled, it will take precedence over the BRAVEHEART keyboard shortcuts, and the keyboard shortcuts for Ctrl + the arrow keys will not work as intended. To disable Mission Control keyboard shortcuts go to **System Preferences** > **Keyboard** > **Shortcuts** > **Mission Control** and disable the relevant shortcuts. The Mission Control shortcuts can be re-enabled in the same way.

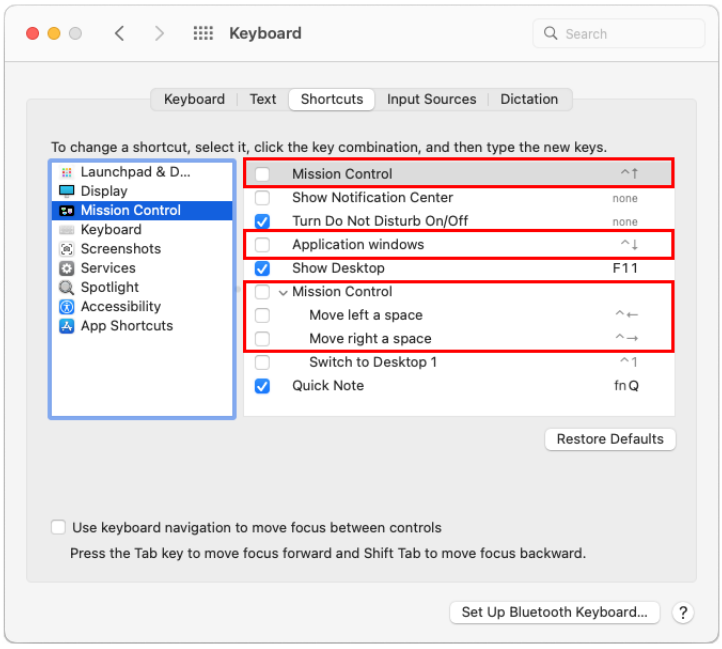


Figure 73: Uncheck the checkboxes outlined in red to allow use of keyboard shortcuts involving the arrow keys. Note that your Mission Control settings appearance may vary slightly depending on your version of Mac OS

27 Equations

Area Vectors:

Area vectors are obtained by taking the area under the relevant segment of the median VCG ($\mathbf{V}(t) = [X(t), Y(t), Z(t)]$) QRST complex using the trapezoidal rule:

$$\mathbf{QRS}_{\text{area}} = \int_{Q_{\text{on}}}^{Q_{\text{off}}} \mathbf{V}(t)\, dt = \left[\int_{Q_{\text{on}}}^{Q_{\text{off}}} X(t)\, dt, \int_{Q_{\text{on}}}^{Q_{\text{off}}} Y(t)\, dt, \int_{Q_{\text{on}}}^{Q_{\text{off}}} Z(t)\, dt\right] \tag{1}$$

$$\mathbf{T}_{\text{area}} = \int_{Q_{\text{off}}}^{T_{\text{off}}} \mathbf{V}(t)\, dt = \left[\int_{Q_{\text{off}}}^{T_{\text{off}}} X(t)\, dt, \int_{Q_{\text{off}}}^{T_{\text{off}}} Y(t)\, dt, \int_{Q_{\text{off}}}^{T_{\text{off}}} Z(t)\, dt\right] \tag{2}$$

Peak Vectors:

Peak vectors are obtained by taking the value of relevant segment of $\mathbf{V}$ at the time point which is of maximum distance from the origin:

$$\mathbf{QRS}_{\text{peak}} = [X(t_{Qmax}), Y(t_{Qmax}), Z(t_{Qmax})] \tag{3}$$

$$\mathbf{T}_{\text{peak}} = [X(t_{Tmax}), Y(t_{Tmax}), Z(t_{Tmax})] \tag{4}$$

where t_{Qmax} is the time of max distance of the QRS loop from the origin, and t_{Tmax} is the time of max distance of the T loop from the origin. These times correspond to the maximum values of the QRS complex and T wave in the VM lead, respectively.

Spatial Ventricular Gradient:

The Spatial Ventricular Gradient (SVG) is the vector created by the QRST integrals in X, Y, and Z:

$$\mathbf{SVG} = \mathbf{QRS}_{\text{area}} + \mathbf{T}_{\text{area}} = \int_{Q_{\text{on}}}^{T_{\text{off}}} \mathbf{V}(t)dt = \left[\int_{Q_{\text{on}}}^{T_{\text{off}}} X(t)\, dt, \int_{Q_{\text{on}}}^{T_{\text{off}}} Y(t)\, dt, \int_{Q_{\text{on}}}^{T_{\text{off}}} Z(t)\, dt\right] \tag{5}$$

Vector Magnitude:

Vector magnitude (VM) is defined as the Euclidean norm of the VCG:

$$\text{VM} = \sqrt{X^2 + Y^2 + Z^2} \tag{6}$$

Azimuth Angle:

Azimuth is defined as the angle in the transverse (XZ) plane with negative angles pointing anterior, and positive angles pointing posterior (see **Figure 71**). Azimuth can take values from 0 to ±180 degrees, with 0 degrees pointing to towards the left, and ±180 degrees pointing towards the right.

$$\text{azimuth} = \arctan\left(\frac{Z}{X}\right) \tag{7}$$

Elevation Angle:

Elevation is defined as the angle up from pointing straight down towards the ground (see **Figure 71**). Values range from 0 to 180 degrees, with 0 degrees pointing towards the feet and 180 degrees pointing towards the head.

$$\text{elevation} = \arccos\left(\frac{Y}{\text{VM}}\right) \tag{8}$$

Absolute Integrals:

The sum absolute integral (SAI) is defined as the area under the absolute value of the QRST complex:

$$\text{SAI}_i = \int_{Q_{\text{on}}}^{T_{\text{off}}} |V_i(t)|\, dt \qquad \text{for } i = X, Y, Z, \text{ or VM} \tag{9}$$

SAI QRST is defined as:

$$\text{SAI QRST} = \text{SAI}_x + \text{SAI}_y + \text{SAI}_z = \int_{Q_{\text{on}}}^{T_{\text{off}}} |X(t)|\, dt + \int_{Q_{\text{on}}}^{T_{\text{off}}} |Y(t)|\, dt + \int_{Q_{\text{on}}}^{T_{\text{off}}} |Z(t)|\, dt \tag{10}$$

QRST Angles:

The spatial QRST angle is the 3-dimensional angle between QRS and T vectors:

$$\text{QRST Angle} = \arccos\left(\frac{\mathbf{QRS} \cdot \mathbf{T}}{|\mathbf{QRS}|\,|\mathbf{T}|}\right) \tag{11}$$

where the peak QRST angle uses $\mathbf{QRS}_{\text{peak}}$ and $\mathbf{T}_{\text{peak}}$, and the area QRST angle (also known as mean QRST angle) uses $\mathbf{QRS}_{\text{area}}$ and $\mathbf{T}_{\text{area}}$.

Total Cosine R to T (TCRT):

TCRT was calculated as previously described using singular value decomposition [15].

VCG Loop Length:

Loop length is calculated as the sum of distances covered as the VCG loop increments by each sample For example, QRS loop length is calculated as:

$$\sum_{i=Q_{\text{on}}}^{Q_{\text{off}-1}} \sqrt{(X_{i+1}-X_i)^2+(Y_{i+1}-Y_i)^2+(Z_{i+1}-Z_i)^2} \tag{12}$$

and T loop length is calculated as:

$$\sum_{i=Q_{\text{off}}}^{T_{\text{off}-1}} \sqrt{(X_{i+1}-X_i)^2+(Y_{i+1}-Y_i)^2+(Z_{i+1}-Z_i)^2} \tag{13}$$

VCG Loop Speed:

The instantaneous speed of the QRS or T loops (v_i) in mV/s for a VCG with frequency f is calculated as the distance traveled in a sample of time:

$$v_i = \sqrt{(X_{i+1}-X_i)^2+(Y_{i+1}-Y_i)^2+(Z_{i+1}-Z_i)^2} \,/\, \Delta t \qquad \text{where } \Delta t = 1000/f \tag{14}$$

Left Ventricular Hypertrophy (LVH):

LVH metrics are calculated as previously described: The Cornell Voltage is calculated as the sum of the S wave in lead V3 + the R wave in lead aVL [16]. The Sokolow-Lyon LVH criteria is calculated as the sum of the S wave in lead V1 and the maximum of the R wave in either lead V5 or V6 [17].

QRS Electrical Axis:

The ECG electrical axis (E_A) in the frontal plane is calculated as:

$$E_A = \arctan\left(\frac{2F}{\sqrt{3}I}\right) \tag{15}$$

where F is the R wave - S wave in the lead aVF median beat, I is the R wave - S wave in the lead I median beat, and arctan is the 2-argument arctangent (results are $-180\,\text{deg}$ to $+180\,\text{deg}$) to assign the positive and negative values correctly based on established conventions. Notably, the factor of $2/\sqrt{3}$ is included to account for variations in signal amplitude when calculating the electrical axis using a combination of a bipolar ECG lead (Lead I) and an augmented unipolar lead (aVF). See [18] for further information.

VCG Loop Morphology and Singular Value Decomposition:

The singular value decomposition (SVD) is used to find the best fit planes for the QRS and T loops separately. Let $X' = X - c_x$, $Y' = Y - c_y$, and $Z' = Z - c_z$ where the centroid $\mathbf{c} = (c_x, c_y, c_z)$ is the mean QRS or T vector. Then let the matrix $\mathbf{M}'$ have the centroid-subtracted leads as columns. The SVD of $\mathbf{M}'$ is:

$$\mathbf{M}' = \mathbf{USV}^T \tag{16}$$

After SVD, unit vectors that span the best fit plane for the VCG loop are given by the columns of $\mathbf{V}$ (right singular vectors).

The 3 singular values along the diagonal of matrix $\mathbf{S}$, when squared, give the proportion of mean-squared error or variance (σ^2) in the direction of the 3 corresponding basis vectors in matrix $\mathbf{V}$:

$$S = \begin{bmatrix} S_1 & 0 & 0 \\ 0 & S_2 & 0 \\ 0 & 0 & S_3 \end{bmatrix} \qquad \sigma^2 = \begin{bmatrix} S_1^2 & 0 & 0 \\ 0 & S_2^2 & 0 \\ 0 & 0 & S_3^2 \end{bmatrix} \tag{17}$$

The columns are arranged such that $S_1 > S_2 > S_3$. Note that if the the VCG loop is perfectly coplanar, then $S_3 = 0$. Assuming that the VCG loop is approximately planar, the third column of matrix $\mathbf{V}$ ($\mathbf{V}_3$) is normal to the best fit plane; let this vector be denoted by $\mathbf{n}$.

The major and minor axes of the VCG loop are given by the first and second columns of $\mathbf{V}$, $\mathbf{V}_1$ and $\mathbf{V}_2$, respectively (the first and second right singular vectors). The signs of the components of $\mathbf{V}$ are arbitrary, and to enforce consistency, we require that the z-component of $\mathbf{V}_3$ ($= \mathbf{n}$) be negative and the x-component of $\mathbf{V}_1$ be positive. To ensure a right-handed coordinate system, $\mathbf{V}_2$ is set to $\mathbf{V}_3 \times \mathbf{V}_1$, which enforces the correct sign convention.

Of note, the squares of the singular values are equivalent to the respective eigenvalues in principle component analysis.

VCG Loop Coplanarity:

The degree of VCG loop coplanarity is assessed in 2 ways. First, the mean squared error (MSE) in the direction normal to the best fit plane (S_3^2) is obtained from squaring the 3rd singular value as noted above. If $S_3^2 = 0$ then all points in the loop are coplanar. Coplanarity is also assessed by calculating RMSE for points in the VCG loop compared to the points projected onto the best fit plane:

For a given point in the QRS or T loops $\mathbf{s} = [s_x, s_y, s_z]$ let $\mathbf{s}' = \mathbf{s} - \mathbf{c}$. Then the projection s'_{proj} relative to the centroid $\mathbf{c}$ onto the best-fit plane is found by subtracting out the component of the vector in the direction of the normal vector $\mathbf{n}$ defined above:

$$\mathbf{s}'_{\text{proj}} = \mathbf{s}' - (\mathbf{s}' \cdot \mathbf{n})\mathbf{n} \tag{18}$$

RMSE is calculated as the square root of the mean squared distance between corresponding points $\mathbf{s}'$ and $\mathbf{s}'_{\text{proj}}$. Using this metric, if all points are coplanar, RMSE = 0.

VCG Loop Dihedral Angle:

VCG loop dihedral angle is defined as the 3-dimensional angle between the unit normal vectors that define the QRS loop best fit plane (N_{QRS}) and the T loop best fit plane (N_{T}). By convention, the dihedral angle is an acute angle:

$$\text{Dihedral Angle} = \arccos(|\mathbf{N}_{\text{QRS}} \cdot \mathbf{N}_{\text{T}}|) \tag{19}$$

VCG Loop Roundness:

Assuming that the MSE in the direction normal to the best fit plane should be relatively small compared to the MSE in the plane itself, the “roundness" (R) of the VCG loop is defined as the ratio of the largest to second largest singular values:

$$R = S_1/S_2 \tag{20}$$

If $R = 1$ the VCG loop is a perfect circle, and as the value of R increases above 1 the loop is more oval or oblong.

VCG Loop Perimeter:

QRS and T loop perimeter are calculated as the length of the QRS or T loop projected into the best fit plane with the set of points (x'_i, y'_i) defined as $\mathbf{s'}_{\text{proj}}$ above, and is different than the QRS and T loop length defined above, which is the length of the loop without projection. The perimeter of the set of points defined by $\mathbf{s'}_{\text{proj}}$ is calculated using Matlab polyshapes.

VCG Loop Area:

QRS and T loop area are calculated as the area of the QRS or T loop projected into the best fit plane with the set of points (x'_i, y'_i) defined as $\mathbf{s'}_{\text{proj}}$ above. The area of the set of points defined by $\mathbf{s'}_{\text{proj}}$ is calculated using Matlab polyshapes which implement the "shoelace formula":

$$A = \frac{1}{2}\left|\sum_{i=1}^{n}(x'_i y'_{i+1} - x'_{i+1}y'_i)\right| \tag{21}$$

where indices are taken modulo n (so $x'_{n+1} = x'_1$ and $y'_{n+1} = y'_1$) to close the loop.

T Wave Mechanical Dispersion (TMD):

T Wave Mechanical Dispersion (TMD) quantifies variation in T wave morphology between ECG leads and is calculated as previously described using singular value decomposition with some slight modifications to signal filtering [19, 15]. In brief, the T wave vector loops are projected into the subspace represented by the first 2 left singular vectors and weighting based on the first 2 singular values. TMD is then calculated as the mean angle (θ_{ij}) between all 21 pairs of independent ECG leads projected into the new subspace $(\mathbf{u_i})$, excluding lead V1:

$$\text{TMD} = \frac{1}{21}\sum_{i\neq j}\theta_{ij} \qquad \text{where } \theta_{ij} = \arccos\left(\frac{\mathbf{u_i}\cdot\mathbf{u_j}}{|\mathbf{u_i}|\,|\mathbf{u_j}|}\right) \qquad \text{for } i,j = \text{I, II, V2, V3, V4, V5, V6} \tag{22}$$

T Wave Residuum (TWR):

T Wave Residuum (TWR) quantifies the non-dipolar components of the T wave with the concept that the energy in the first 3 leads of the decomposed T wave represent the dipolar components, and the small amount of energy contained remaining 5 leads represent the non-dipolar components. There are some variations in how TWR is calculated. BRAVEHEART uses the following equation for the absolute TWR (TWR_{abs}) which is the sum of the squares of the 4th

through 8th singular values (equivalent to the sum of the 4th through 8th eigenvalues) [20, 21]:

$$\mathrm{TWR}_{\mathrm{abs}} = \sum_{i=4}^{8} S_i^2 \tag{23}$$

The relative TWR ($\mathrm{TWR}_{\mathrm{rel}}$) is the percent of the absolute TWR divided by the sum of the squares of the 1st through 8th singular values (equivalent to the sum of the 1st through 8th eigenvalues):

$$\mathrm{TWR}_{\mathrm{rel}} = 100 \times \frac{\sum_{i=4}^{8} S_i^2}{\sum_{i=1}^{8} S_i^2} \tag{24}$$

QRS Loop Rotation Direction:

To determine if a QRS loop is rotating clockwise (CW) or counterclockwise (CCW), a viewing plane is determined as a normal unit vector to a plane, and the QRS points are projected into that plane as noted above. the signed area of the loop is calculated using the "shoelace formula" as noted above in the information on how VCG loop area is calculated. For loop direction, however, the area can be positive or negative:

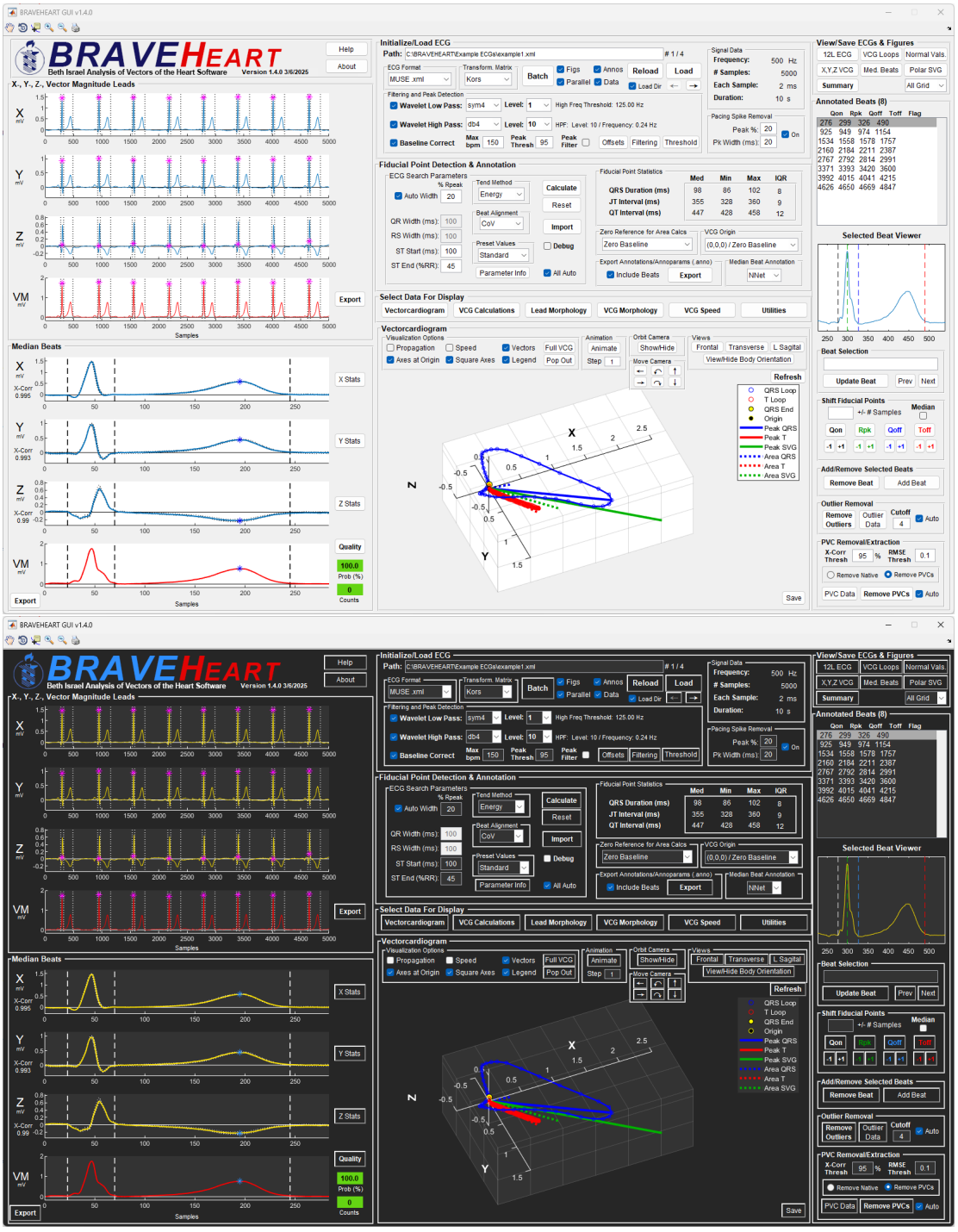
$$A = \frac{1}{2} \sum_{i=1}^{n} (x'_i y'_{i+1} - x'_{i+1} y'_i) \tag{25}$$

where indices are taken modulo n (so $x'_{n+1} = x'_1$ and $y'_{n+1} = y'_1$) to close the loop.

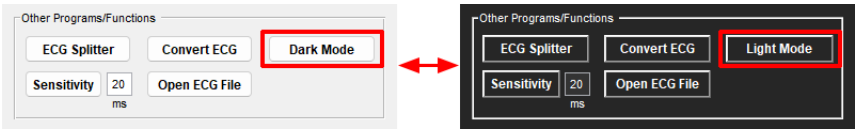
Based on our coordinate system conventions, if $A > 0$ the rotation is "CCW", and if $A < 0$ the rotation is "CW". If the absolute value of the signed area is very small (nominally < 0.1, although this threshold can be adjusted), the rotation is considered "Indeterminate", as it is not clear if the is loop just small, or a figure of 8/crossing over itself with a complex path, in which case it is difficult to define what is "CCW" or "CW". BRAVEHEART reports the QRS loop rotation when viewed in the XY plane (frontal) from the front, the XZ plane (transverse) viewed from below with the patient facing upwards, the ZY plane (left sagital) viewed from the left, and the best fit plane viewed from the direction of the assigned normal vector $\mathbf{n}$. Note that a "CCW" loop will appear "CW" (and vice versa) when viewed from the opposite direction.

28 Light/Dark Mode

As of version 1.4.0, BRAVEHEART allows customization of GUI colors, with the ability to rapidly toggle between "light mode" (the default color scheme), or "dark mode" which nominally is set to have a darker color scheme.



Light/dark mode is toggled in the Utilities section of the GUI (item **6.6.5**) using the Dark Mode or Light Mode buttons:



In light mode and dark mode, color schemes can be customized via the file `check_darkmode.m`:

```
% Colors to use after change of color mode to LIGHT MODE
light_colors.bgcolor = [0.94 0.94 0.94];                % Background of GUI
light_colors.buttoncolor = [0.94 0.94 0.94];            % Button color
light_colors.txtcolor = [0 0 0];                        % Main Text color
light_colors.bgfigcolor= [1 1 1];                       % Figure/graph background
light_colors.xyzecg = [0 0.45 0.74];                    % Color of XYZ ECG signals
light_colors.vmecg = [1 0 0];                           % Color of VM ECG signals
light_colors.vertlines = [0 0 0];                       % Color of vertical graph lines
light_colors.bluetxtcolor = [0 0 1];                    % Color of blue text (if change)
light_colors.pvcmarker = [0, 0.5, 0];                   % Color of PVC marker
light_colors.outliermarker = [0.25 0.25 0.25];          % Color of outlier marker

% Colors to use after change of color mode to DARK MODE
dark_colors.bgcolor= [0.15 0.15 0.15];
dark_colors.buttoncolor = [0.15 0.15 0.15];
dark_colors.txtcolor = [0.85 0.85 0.85];
dark_colors.bgfigcolor = [0.2 0.2 0.2];
dark_colors.xyzecg = [1 0.89 0];
dark_colors.vmecg = [1 0 0]; [1 0.03 0];
dark_colors.vertlines = [1 1 1];
dark_colors.bluetxtcolor = [0.18 0.56 1];
dark_colors.pvcmarker = [0.9 0.9 0.9];
dark_colors.outliermarker = [0 0.91 1];
```

Colors are edited as RBG triplets normalized to a 0-1 range over the usual RBG range of 0-255 (eg [0 255 125] = [0 1 0.49]). Hex color codes will not work.

A description of the light/dark color schemes and the nominal colors are shown below:

Variable	Description	Light RGB & Color	Dark RGB & Color
`.bgcolor`	GUI background	[0.94 0.94 0.94]	[0.15 0.15 0.15]
`.buttoncolor`	Button background	[0.94 0.94 0.94]	[0.15 0.15 0.15]
`.txtcolor`	Main text color	[0 0 0]	[0.85 0.85 0.85]
`.bgfigcolor`	Figure plot color	[0 0 0]	[0.85 0.85 0.85]
`.xyzecg`	X, Y, Z plot color	[0 0.45 0.74]	[1 0.89 0]
`.vmecg`	VM plot color	[1 0 0]	[1 0 0]
`.vertlines`	Vertical graph lines	[0 0 0]	[1 1 1]
`.bluetxtcolor`	Blue text color	[0 0 1]	[0.18 0.56 1]
`.pvcmarker`	PVC marker color	[0 0.5 0]	[0.9 0.9 0.9]
`.outliermarker`	Outlier marker color	[0.25 0.25 0.25]	[0 0.91 1]

Some colored GUI elements, such as the logo and green, pink, or orange text, are not affected by light or dark mode. Some figures will not respect the light/dark mode color for figure background (`.bgfigcolor`) because it was decided that a white background was optimal for data display. There are some operating system specific idiosyncrasies that may also make some GUI elements (such as dropdowns) display differently on different operating systems regardless of what colors are selected.

Of note, the GUI color scheme always defaults to the the "light" (grey) color scheme when it is loaded for the first time. These colors are the nominal colors settings for light mode, but if you change the light mode colors, you will have to cycle to dark mode and then back to light mode to see your changes.

When you switch between light and dark mode, please make sure all external figures are closed to avoid any unexpected behavior.

29 Annotation Parameter Links with GUI

This section contains the links between parameters in `Annoparams.m` and where the parameter value is editable in the BRAVHEART GUI. (NA = Parameter cannot be adjusted via GUI)

- `maxBPM` ⟷ GUI item **1.6** textbox
- `pkthresh` ⟷ GUI item **1.7** textbox
- `pkfilter` ⟷ GUI item **1.8** checkbox
- `lowpass` ⟷ GUI item **1.3** checkbox
- `highpass` ⟷ GUI item **1.4** checkbox
- `wavelet_level_lowpass` ⟷ GUI item **1.3** dropdown
- `wavelet_level_highpass` ⟷ GUI item **1.4** dropdown
- `wavelet_name_lowpass` ⟷ GUI item **1.3** dropdown
- `wavelet_name_highpass` ⟷ GUI item **1.4** dropdown
- `baseline_correct_flag` ⟷ GUI item **1.5** checkbox
- `transform_matrix_str` ⟷ GUI item **1.2** dropdown
- `baseline_flag` ⟷ GUI item **3.12** dropdown
- `origin_flag` ⟷ GUI item **3.13** dropdown
- `autoMF` ⟷ GUI item **3.1** checkbox
- `autoMF_thresh` ⟷ GUI item **3.1** textbox
- `MF_width` ⟷ NA
- `QRwidth` ⟷ GUI item **3.2** textbox
- `RSwidth` ⟷ GUI item **3.2** textbox
- `STstart` ⟷ GUI item **3.2** textbox
- `STend` ⟷ GUI item **3.2** textbox
- `spike_removal` ⟷ GUI item **1.18** checkbox
- `pacer_spike_width` ⟷ GUI item **1.18** textbox
- `pacer_mf` ⟷ NA
- `pacer_thresh` ⟷ GUI item **1.18** textbox
- `cwt_spike_removal` ⟷ GUI item **1.18** checkbox
- `interpolate` ⟷ GUI item **1.18** checkbox
- `pacer_zcut` ⟷ GUI item **1.18** textbox
- `pacer_zpk` ⟷ GUI item **1.18** textbox
- `pacer_maxscale` ⟷ GUI item **1.18** textbox
- `pacer_spike_num` ⟷ GUI item **1.18** textbox
- `align_flag` ⟷ GUI item **3.4** dropdown
- `cov_mf` ⟷ NA
- `cov_thresh` ⟷ NA
- `shiftq` ⟷ NA
- `shiftt` ⟷ NA
- `Tendstr` ⟷ GUI item **3.3** dropdown
- `median_reanno_method` ⟷ GUI item **3.15** dropdown
- `outlier_removal` ⟷ GUI item **9.4** checkbox

- `modz_cutoff` ⟷ GUI item **9.3** textbox
- `pvc_removal` ⟷ GUI item **9.10** checkbox
- `pvcthresh` ⟷ GUI item **9.5** textbod
- `rmse_pvcthresh` ⟷ GUI item **9.6** textbox
- `keep_pvc` ⟷ GUI item **9.7** radio button
- `blanking_window_q` ⟷ GUI item **6.5.1** textbox
- `blanking_window_t` ⟷ GUI item **6.5.2** textbox
- `debug` ⟷ GUI item **3.10** checkbox

30 Testing Framework

It is important to be certain that any changes to the BRAVEHEART source code have not broken or otherwise adversely affected the main functions of the software, including ECG processing/filtering, annotation, PVC/outlier removal, manual beat removal/editing, and calculations. In order to ensure that any changes have not inadvertently affected critical parts of the BRAVEHEART source code, we have provided a testing framework that can be used to validate any changes that are made. The testing framework is located in `test_braveheart.m`. This is a very long file that contains a multitude of tests designed to determine if there are any errors in any parts of the BRAVEHEART ECG/VCG processing pipeline.

The testing framework can be run by opening the `test_braveheart.m` file and clicking the "Run Tests" icon in the Editor toolbar, or by typing `runtests( "test_braveheart" )` in the MATLAB command line. The testing functions will run and at the end you will get a report about how many functions passed or failed. If the software is functioning properly, all tests should pass. If a test fails, MATLAB provides information on the error that can be used to troubleshoot.

If you make changes to some of the BRAVEHEART core functions such as signal filtering or annotation, it is possible that the tests may fail even though BRAVEHEART is functioning properly within the parameters of this new code. This is because functions like filtering and annotation affect all downstream processing and measurements. The testing framework assumes that the **current** filtering and annotation functions are being used, and the measurements that are checked as part of testing will therefore generate errors if alternate filtering/annotation functions are used. If such changes are made, the testing framework expected outputs can be updated.

The testing framework will not automatically update if new parameters are added to the BRAVEHEART results classes (see **Chapter 10**), but adding new parameters will not break the testing framework.

The BRAVEHEART GitHub automatically runs the testing framework for all commits to ensure that the most up-to-date version of code in the Github repository will function accurately and without errors.

31 Troubleshooting/Help/FAQs

31.1 The GUI is Cut Off/Missing Some Features

If the GUI is missing some features when compared to **Figure 1** when you first open the BRAVEHEART GUI, the most likely cause is that your monitor resolution is too low. Adjust your monitor resolution so that it is atleast 1920 x 1080, and then reload the GUI. Note that there are minor differences in the appearance of GUI elements depending on your operating system.

If your screen resolution is 1920 x 1080 or greater and the GUI is still not fully displayed, it is likely that your computer display settings have some form of scaling turned on; this setting increases the size of text etc but also effectively reduces the screen resolution.

To disable screen scaling:
Windows 10: Go to the **Ease of Access** settings by pressing the Windows key + U. Under **Make everything bigger** on the **Display** tab, change to 100%.

Windows 11: Open **Settings** and then **Display**. Under **Scale & layout**, expand the **Scale** menu and change to 100%.

Mac OS: Open **System Preferences** and then **Display**. Choose **Scaled** Resolution and then **More Space**.

31.2 The GUI Looks Very Strange When Using MATLAB R2025a or Later

MATLAB R2025a introduced a major graphics overhaul and formal removal of GUIDE GUI support. If you run a version of BRAVEHEART prior to 1.6.0 in R2025a, it will look disorganized and chaotic due to these changes. If you are using MATLAB R2025a please make sure your BRAVEHEART version is 1.6.0 or later. Alternatively, you can always use an older version of MATLAB prior to R2025a which will not have this issue.

31.3 My ECG Won't Load

If an ECG won't load, first check that the correct file format is selected, and try to load another ECG of the same format to ensure that the format is set correctly.

The most common reason for an ECG not loading is significant noise or artifact that interferes

with QRS peak localization (because QRS peak localization depends on thresholding the full VM signal, large, focal artifacts can result in almost all of the ECG being below the percentile value set by `pkthresh`). By enabling ☑ Safe Mode (see **Chapter 31.4** below) in the GUI **Utilities** section (GUI item **6.6.7**), after the ECG signals are loaded you can view the 12-lead ECG by clicking on the 12L ECG (GUI item **7.1**). This may reveal a large area of noise/artifact, or significant baseline wander.

The first thing to try is to decrease the value of `pkthresh` by lowering the value in the 'Peak Threshold' textbox in GUI section **1.6** to a very low value around 10. In fact, if the ECG throws an error but loads in Safe Mode (GUI item **6.6.7**), without changing any other settings, the value of `pkthresh` is likely the cause of the error, as Safe Mode also automatically lowers `pkthresh` to 10 in addition to disabling processing that occurs after filtering. If this resolves the error, you may need to increase `pkthresh` somewhat to avoid overcalling QRS peaks. Use of the Threshold button (GUI item **1.15** – see **Chapter 15.1**, **Figure 24**, **Figure 25**) can help visualize what the QRS peak detector is doing.

If adjusting `pkthresh` does not resolve the issue, next try to lower the the high-pass filtering level (GUI item **1.4**) by 1-2 levels. This often helps remove any low-frequency wander by increasing the high-pass filter cutoff frequency. However, lowering the high-pass filtering level (and therefore increasing the high-pass filter cutoff frequency) too much can adversely affect measurements by distorting the QRST complex. If filtering is removing frequencies between 0.5-40 Hz there will be some concern that the QRST complex may be distorted by filtering [3].

If there is a focal section of the ECG that is causing errors, another option is to load the ECG in Safe Mode and then use the **ECG Splitter** button/function in the **Utilities** section (GUI item **6.6.5**). The abnormal area of the ECG can be cut out and the remainder of the ECG without the severe artifact can be analyzed. It should be noted that using a shortened ECG will likely affect the measured values slightly because baseline correction depends on the calculated HR.

31.4 Safe Mode

Safe mode can help in cases where an ECG will not load. As noted above, this is often due to significant noise and artifact that cause problems with initial R peak detection. Clicking 'Safe Mode' in GUI item **6.6.7** disables all processing after filtering and decreases the value of `pkthresh` to 10 to increase the chances that the ECG can load without an error. This can allow you to visualize the ECG/VCG using GUI section 7.

31.5 The ECG Loads but I get an Error When Press Calculate

This is usually due to an issue with first pass or median beat annotation. Attempt to troubleshoot this issue in the same was as shown above when dealing with ECGs that won't load. Checking the 'Debug' figures (see **Chapter 31.8**) can also help.

31.6 A Measured Parameter Returns as NaN

This error is usually due to noise/artifact which can be dealt with as noted above. The other reason that this may occur is if a median beat fiducial point was not found. This most commonly occurs if the median beat annotater did not find a T_{off}. This can occur if the ECG is very noisy, or in good quality ECGs if the QT interval is long or if the QRS morphology is atypical to the point where the QRS width and/or QT search windows are too short. If this occurs, using the ☑ Debug option (see **Chapter 31.8**) can help troubleshoot, and often adjusting the annotation parameters will allow the annotater to find all fiducial points. In rare occasions when the neural network median beat annotater fails but the ECG quality is reasonable, switching `median_reanno_method` from `'NNet'` to `'Std'` may be useful. The overall accuracy of the `'Std'` median beat annotater is significantly lower than the `'NNet'` median beat annotater, especially for low amplitude T waves or atypical QRST morphologies.

31.7 Error Logs

When using the GUI, under Utilities there is an option for ☑ Error Logging (GUI item 6.6.3). If this is checked, loading an ECG will generate a text file called `braveheart_error_log_<date_time>.txt` which includes information on the file, annotations parameters, and any errors that are generated. Note that logging is started during ECG loading, so to start logging errors you will need to load/reload the ECG after checking ☑ Error Logging. Starting in v1.1.0, when running via executable, the MATLAB console is logged to a file `logfile.txt`. We are happy to assist with interpretation of error logs if needed - contact us at `braveheart.ecg@gmail.com`.

31.8 Debugging Annotation Figures

The ☑ Debug checkbox (GUI item 4.9) enables graphical display of both first pass and median beat annotation, and can be helpful when troubleshooting ECGs that cause errors or do not annotate properly. Debug figures can help determine how to adjust annotation parameters to allow the ECG to successfully process. When ☑ Debug is checked and Calculate is pressed, figures showing annotation windows for first pass annotation and neural network predicted

fiducial point probabilities are displayed. See **Chapter 18**, **Chapter 18.3**, **Figure 45**, and **Figure 70** for further details. These figures can often help you determine if adjustment of annotation parameters will resolve any errors.

31.9 How should I cite use of the BRAVEHEART software?

Please cite our Open Access methods manuscript [1] which is currently available in *Computer Methods and Programs in Biomedicine*:

Hans F. Stabenau and Jonathan W. Waks, BRAVEHEART: Open-source software for automated electrocardiographic and vectorcardiographic analysis. *Computer Methods and Programs in Biomedicine* 2023 Dec:242:107798; DOI: `https://doi.org/10.1016/j.cmpb.2023.107798`.

31.10 Other Frequently Asked Questions

1. Can BRAVEHEART use scanned ECGs in `.jpg` or `.pdf` format?

No – BRAVEHEART requires ECG data to be in some raw, digitized format. The formats that are included with BRAVEHEART are described in **Chapter 23**, and new formats can be added as needed as described in **Chapter 8**. Construction of VCGs also requires simultaneous recording of the 8 independent ECGs leads (I, II, V1–V6), but if you have printed ECGs of decent quality that have simultaneous rhythm strips of at least the 8 independent leads, consider using a program to digitize your ECG [22] and then import the digitized version into BRAVEHEART.

2. Why are the high-pass and low-pass wavelet levels not automatically adjusting when I run BRAVEHEART via the command line like they do when I use the GUI?

The wavelet levels only adjust based on the ECG sampling frequency when using the GUI. When using the command line version of BRAVEHEART (`braveheart_batch.m`) you must set the high- and low-pass wavelet levels manually in `Annoparams.m`. Details about how to adjust the wavelet levels to adjust the filtering cutoff frequencies are available in **Chapter 7** and **Table 1**.

3. Can BRAVEHEART annotate or measure parameters from P waves?

Not currently, although we are in the process of adding P wave annotation which will allow assessment of P wave parameters. P wave annotation/processing is more complex than QRST annotation due to the fact that a cardiac cycle always has a single QRST complex in a predictable order, but atrial activity can be completely dissociated from ventricular activity (or absent completely) and, in general, P waves are lower amplitude signals with a lower signal to noise ratio.

4. I don't have MATLAB, but I need to add a new ECG format, transformation matrix, or parameter. What can I do?

Contact us at `braveheart.ecg@gmail.com` and we will try to assist you.

5. I found a bug/error. How can I tell you whats wrong?

Contact us at `braveheart.ecg@gmail.com` with as many details as possible (and possibly an error log – see **Chapter 31.7**) and we will try to resolve the issue.

6. How can I change the default parameters so I don't have to adjust them each time I use the program?

The default parameters for use in both the command line and GUI versions of

BRAVEHEART are controlled by setting values within `Annoparams.m` when running via MATLAB, or `Annoparams.csv` when running via executable. Edit and save the values in these files to control the default values that appear when the GUI is opened or the command line version is run (see **Chapter 6**). NOTE: there are some parts of the GUI that operate based on the specific ECG format chosen (see **Chapter 8**). This includes choosing specific wavelet decomposition levels for specific ECG formats based on ECG sampling frequency. Once the GUI is loaded, choosing a new ECG format may adjust the denoising/filtering settings. If these changes based on specific ECG formats are causing issues, you can edit them in the file `ecg_source_gui.m`. These settings are ONLY used with the GUI.

7. I have an ECG in a format that BRAVEHEART should be able to read, but it's not working. Whats wrong?

It is possible that your file is a variant/different version of the formats that can be read by BRAVEHEART, as unfortunately we were not able to test every possible version of each included format. You can look through the code in the appropriate load module `.m` file (see **Chapter 23**) to see if you can find the issue and modify the code. If you can't edit the code or are having issues, please email us at `braveheart.ecg@gmail.com` and we can help troubleshoot. If possible, please send us a de-identified copy of the file that is not working.

8. I have ECG data that is not in a 12-lead ECG format (eg a single lead or a subset of leads). Can BRAVEHEART analyze these data?

BRAVEHEART was designed to process 12-lead ECGs and do VCG calculations. However, it is possible to analyze subsets of ECG lead data in a limited way. BRAVEHEART requires 12-lead ECGs to generate meaningful VCGs and calculation of VCG parameters, but if you only want to measure parameters on a single lead (such as the parameters included in the `Lead_Morphology` results class), BRAVEHEART can analyze more limited ECG lead datasets. At the present time there is no way to directly load a single lead or subset of leads into BRAVEHEART, but we are working on including this functionality in the future. For now, to load a single lead or a subset of leads, you need to duplicate the available data so that BRAVEHEART has 12 leads to work with. There are different ways of doing this within the source code if you have MATLAB and can edit `ECG12.m`. If you don't have access to MATLAB and are running via executable, the best way is to duplicate your data within the raw data files and then load via whatever ECG format works best. Keep in mind that if you do not supply a valid 12-lead ECG, all VCG measurements will be unreliable and should not be used.

9. Can BRAVEHEART be used for heart rate variability (HRV) analyses?

BRAVEHEART was designed for 12-lead ECG/VCG analysis. HRV analyses are usually performed on long term continuous ECG or Holter recordings which are usually not 12-lead ECG data, and which are usually significantly longer than 12-lead ECG data (hours to days vs seconds to minutes). As of version 1.2.2 BRAVEHEART does calculate some simple HRV metrics such as standard deviation of normal-to-normal R-R intervals (SDNN) and root mean square of successive differences (RMSSD) (see **Chapter 24.4**). A full HRV implementation is not planned at this point in time. Users who need more detailed HRV analyses can find one of the many HRV software packages that are available such as HRVTool (`https://github.com/MarcusVollmer/HRV`). As of version 1.2.2 BRAVEHEART can export the timing of RR intervals for use in other software packages by exporting the processed ECG data as a .mat file and looking at the data stored in `data.beats` (see **Chapter 10.1.1**).

10. I don't have MATLAB. Can BRAVEHEART run on Octave?

BRAVEHEART will not run on Octave. If you do not have MATLAB you can run the compiled executable version of BRAVEHEART (see **Chapter 4.2**).

32 External Code/Files

BRAVEHEART uses some existing open-source code which includes:

- draggable – Francois Bouffard (2023)
  `https://www.mathworks.com/matlabcentral/fileexchange/4179-draggable`,
  MATLAB Central File Exchange. Retrieved February 5, 2020.
- LZW Compression Algorithm – Giuseppe Ridinò (2023)
  `https://www.mathworks.com/matlabcentral/fileexchange/4899-lzw-compression-algorithm`,
  MATLAB Central File Exchange. Retrieved June 1, 2022.
- Sierra ECG Tools – David D. Salcido and Christopher A. Watford (2014).
  `https://github.com/sixlettervariables/sierra-ecg-tools`,
  Retrieved June 1, 2022.
- The BioSig Project for MATLAB v3.8.4 – Alois Schloegl (2024)
  `https://https://biosig.sourceforge.net/`, Retrieved March 1, 2024.

33 Publications Using BRAVEHEART

The following publications have utilized BRAVEHEART for ECG/VCG analysis.

[1] H. F. Stabenau, C. Shen, L. G. Tereshchenko, and J. W. Waks. Changes in global electrical heterogeneity associated with dofetilide, quinidine, ranolazine, and verapamil. *Heart Rhythm*, 2020 Mar;17(3):460-467.

[2] H. F. Stabenau, C. Shen, P. Zimetbaum, A. E. Buxton, L. G. Tereshchenko, and J. W. Waks. Global electrical heterogeneity associated with drug-induced torsades de pointes. *Heart Rhythm*, 2021 Jan;18(1):57-62.

[3] H. F. Stabenau, C. P. Bridge, and J. W. Waks. ECGAug: A novel method of generating augmented annotated electrocardiogram QRST complexes and rhythm strips. *Comput Biol Med*, 2021 Jul;134:104408.

[4] H. F. Stabenau, M. Marcus, J. D. Matos, I. McCormick, D. Litmanovich, W. J. Manning, B. J. Carroll, and J. W. Waks. The spatial ventricular gradient is associated with adverse outcomes in acute pulmonary embolism. *Ann Noninvasive Electrocardiol*, 2023, Jan 24;e13041.

[5] A. N. Rosas Diaz, H. F. Stabenau, G. P. Hurtado, S. Warack, J. W. Waks, and A. Asnani. The spatial ventricular gradient is an independent predictor of anthracycline-associated cardiotoxicity. *JACC: Adv*, 2(2):100269, 2023.

[6] H. F. Stabenau, A. Sau, D. B. Kramer, N. S. Peters, F. S. Ng, and J. W. Waks. Limits of the Spatial Ventricular Gradient and QRST Angles in Patients with Normal Electrocardiograms and No Known Cardiovascular Disease Stratified by Age, Sex, and Race. *J Cardiovasc Electrophysiol*, 2023 Nov;34(11):2305-2315.

[7] N. Isaza, H. F. Stabenau, D. B. Kramer, A. Sau, P. Tung, T. R. Maher, A. H. Locke, P. Zimetbaum, A. d'Avila, N. S. Peters, L. G. Tereshchenko, F. S. Ng, A. E. Buxton, and J. W. Waks. The Spatial Ventricular Gradient is Associated with Inducibility of Ventricular Arrhythmias During Electrophysiology Study. *Heart Rhythm*, 2024 Nov;21(11):2160-2167.

[8] L. Pastika, A. Sau, K. Patlatzoglou, E. Sieliwonczyk, A. H. Ribeiro, K. A. McGurk, S. Khan, D. Mandic, W. R. Scott, J. S. Ware, N. S. Peters, A. L. P. Ribeiro, D. B. Kramer, J. W. Waks, and F .S. Ng. Deep Neural Network-derived Electrocardiographic Body Mass Index as a Predictor of Cardiometabolic Disease. *NPJ Digit Med*, 2024 Jun 25;7(1):167.

[9] A. Sau, L. Pastika, E. Sieliwonczyk, K. Patlatzoglou, A. H. Ribeiro, K. A. McGurk, B. Zeidaabadi, H. Zhang, K. Macierzanka, D. Mandic, E. Sabino, L. Giatti, S. M. Barreto, L. do Valle Camelo, I. Tzoulaki, D. P. O'Regan, N. S. Peters, J. S. Ware, A. L. P. Ribeiro, D. B. Kramer, J. W. Waks, and F. S. Ng. Artificial Intelligence–Enabled Electrocardiogram for Mortality and Cardiovascular Risk Estimation: An Actionable, Explainable and Biologically Plausible Platform. *Lancet Digit Health*, 2024 Nov;6(11):e791-e802.

[10] M. Raad, D. B. Kramer, H. F. Stabenau, E. Anyanwu, D. S. Frankel, and J. W. Waks. The Spatial Ventricular Gradient Is Associated with Pacing-Induced Cardiomyopathy. *Heart Rhythm*, 2024. Dec 28:S1547-5271(24)03710-X.

[11] A. Sau, J. Barker, L. Pastika, E. Sieliwonczyk, K. Patlatzoglou, K. A. McGurk, N. S. Peters, D. P. O'Regan, J. S. Ware, D. B. Kramer, J. W. Waks, and F. S. Ng. Artificial Intelligence-Enhanced Electrocardiography for Prediction of Incident Hypertension. *JAMA Cardiol*, 2025 Mar;10(3):214-223.

[12] K. Macierzanka, A. Sau, K. Patlatzoglou, L. Pastika, E. Sieliwonczyk, M. Gurnani, N. S. Peters, J. W. Waks, D. B. Kramer, and F. S. Ng. Siamese neural network-enhanced electrocardiography can re-identify anonymized healthcare data. *Eur Heart J Digit Health*, 2025 May;6(3):417-426.

[13] A. Sau, E. Sieliwonczyk, K. Patlatzoglou, L. Pastika, K. McGurk, A. H. Ribeiro, A. L. P. Ribeiro, J. E. Ho, N. S. Peters, J. S. Ware, U. Tayal, D. B. Kramer, J. W. Waks, and F. S. Ng. Artificial intelligence-enhanced electrocardiography for the identification of a sex-related cardiovascular risk continuum: a retrospective cohort study. *Lancet Digit Health*, 2025 Mar;7(3):e184-e194.

[14] M. Gurnani, K. Patlatzoglou, J. Barker, D. Bivona, L. Pastika, E. Sieliwonczyk, B. Zeidaabadi, P. Inglese, L. Curran, A. D. Arnold, D. O'Regan, Z. Whinnett, K. C. Bilchick, N. S. Peters, D. B. Kramer, J. W. Waks, A. Sau, and F. S. Ng. Revisiting Abnormalities of Ventricular Depolarization: Redefining Phenotypes and Associated Outcomes Using Tree-Based Dimensionality Reduction. *J Am Heart Assoc*, 2025 Jun 18:e040814.

[15] Y. Liang, A. Sau, B. Zeidaabadi, J. Barker, K. Patlatzoglou, L. Pastika, E. Sieliwonczyk, Z. Whinnett, N. S. Peters, Z. Yu, X. Liu, S. Wang, H. Lu, D. B. Kramer, J. W. Waks, Y. Su, J. Ge, and F. S. Ng. Artificial intelligence-enhanced electrocardiography to predict regurgitant valvular heart diseases: an international study. *Eur Heart J*, 2025 Jul 16:ehaf448.

[16] A. J. Shepherd, H. F. Stabenau, A. Sau, P. Tung, T. R. Maher, S. Yang, A. H. Locke, P. Zimetbaum, G. F. Michaud, A. d'Avila, N. S. Peters, A. E. Buxton, F. S. Ng, D. B. Kramer, and J. W. Waks. Larger Spatial Ventricular Gradient Magnitude is Associated with Higher Rates of Response to Cardiac Resynchronization Therapy. *Heart Rhythm O2*, 2025 In Press.

[17] A. Sau, H. Zhang, J. Barker, L. Pastika, K. Patlatzoglou, B. Zeidaabadi, A. El-Medany, G. R. Khattak, K. A. McGurk, E. Sieliwonczyk, J. S. Ware, N. S. Peters, D. B. Kramer, J. W. Waks, and, F. S. Ng. Artificial Intelligence-Enhanced Electrocardiography for Complete Heart Block Risk Stratification. *JAMA Cardiol*, 2025 Aug 20:e252522.

[18] A. Sau, E. Sieliwonczyk, J. Barker, B. Zeidaabadi, L. Pastika, K. Patlatzoglou, G. R. Khattak, K. A. McGurk, N. S. Peters, D. B. Kramer, J. W. Waks, J. S. Ware, and F. S. Ng. Prediction of incident atrial fibrillation: a comprehensive evaluation of conventional and AI-enhanced approaches. *Heart Rhythm*, 2025 In Press.

[19] H. F. Stabenau, J. D. Matos, D. B. Kramer, A. Sau, L. Pastika, E. Sieliwonczyk, K. Patlatzoglou, N. S. Peters, P. Tung, F. S. Ng, R. Nezafat, and J. W. Waks. The Spatial Ventricular Gradient is a Non-Invasive, Vectorcardiographic Correlate of Cardiac Fibrosis. *J Interv Card Electrophysiol*, 2025. In Press

34 References

[1] H. F. Stabenau and J. W. Waks. Braveheart: Open-source software for automated electrocardiographic and vectorcardiographic analysis. *Comput Methods Programs Biomed*, Dec:242:107798, 2023.

[2] H. F. Stabenau, A. Sau, D. B. Kramer, N. S. Peters, F. S. Ng, and J. W. Waks. Limits of the spatial ventricular gradient and qrst angles in patients with normal electrocardiograms and no known cardiovascular disease stratified by age, sex, and race. *J Cardiovasc Electrophysiol*, Nov;34(11):2305-2315, 2023.

[3] N. V. Thakor, J. G. Webster, and W. J. Tompkins. Estimation of QRS complex power spectra for design of a QRS filter. *IEEE Trans Biomed Eng*, 31(11):702–706, Nov 1984.

[4] J. A. Kors, G. van Herpen, A. C. Sittig, and J. H. van Bemmel. Reconstruction of the Frank vectorcardiogram from standard electrocardiographic leads: diagnostic comparison of different methods. *Eur Heart J*, 11(12):1083–1092, Dec 1990.

[5] G. E. Dower, H. B. Machado, and J. A. Osborne. On deriving the electrocardiogram from vectoradiographic leads. *Clin Cardiol*, 3(2):87–95, Apr 1980.

[6] L. Johannesen, J. Vicente, M. Hosseini, and D. G. Strauss. Automated Algorithm for J-Tpeak and Tpeak-Tend Assessment of Drug-Induced Proarrhythmia Risk. *PLoS One*, 11(12):e0166925, 2016.

[7] B. Iglewicz and D. Hoaglin. *The ASQC Basic References in Quality Control: Statistical Techniques*, chapter Volume 16: How to Detect and Handle Outliers. 1993.

[8] P. S. Addison. Wavelet transforms and the ECG: a review. *Physiol Meas*, 26(5):R155–199, Oct 2005.

[9] P. Kligfield, L. S. Gettes, J. J. Bailey, R. Childers, B. J. Deal, E. W. Hancock, G. van Herpen, J. A. Kors, P. Macfarlane, D. M. Mirvis, O. Pahlm, P. Rautaharju, and G. S. Wagner. Recommendations for the standardization and interpretation of the electrocardiogram. *Circulation*, 115(10):1306–1324, 2007.

[10] H. F. Stabenau, C. Shen, L. G. Tereshchenko, and J. W. Waks. Changes in global electrical heterogeneity associated with dofetilide, quinidine, ranolazine, and verapamil. *Heart Rhythm*, 17(3):460–467, 03 2020.

[11] Recommendations for measurement standards in quantitative electrocardiography. The CSE Working Party. *Eur Heart J*, 6(10):815–825, Oct 1985.

[12] S. H. Zhou, E. D. Helfenbein, J. M. Lindauer, R. E. Gregg, and D. Q. Feild. Philips QT interval measurement algorithms for diagnostic, ambulatory, and patient monitoring ECG applications. *Ann Noninvasive Electrocardiol*, 14 Suppl 1:3–8, Jan 2009.

[13] J. L. Willems, P. Arnaud, J. H. van Bemmel, P. J. Bourdillon, C. Brohet, S. Dalla Volta, J. D. Andersen, R. Degani, B. Denis, and M. Demeester. Assessment of the performance of electrocardiographic computer programs with the use of a reference data base. *Circulation*, 71(3):523–534, Mar 1985.

[14] C. Zywietz and D. Celikag. Testing results and derivation of minimum performance criteria for computerized ecg-analysis. In *[1991] Proceedings Computers in Cardiology*, pages 97–100, 1991.

[15] B. Acar, G. Yi, K. Hnatkova, and M. Malik. Spatial, temporal and wavefront direction characteristics of 12-lead T-wave morphology. *Med Biol Eng Comput*, 37(5):574–584, Sep 1999.

[16] P. M. Okin, M. J. Roman, R. B. Devereux, and P. Kligfield. Electrocardiographic identification of increased left ventricular mass by simple voltage-duration products. *Journal of the American College of Cardiology*, 25(2):417–423, 1995.

[17] M. Sokolow and T. P. Lyon. The ventricular complex in left ventricular hypertrophy as obtained by unipolar precordial and limb leads. *American Heart Journal*, 37(2):161–186, 1949.

[18] D. Novosel, G. Noll, and T. F. scher. Corrected formula for the calculation of the electrical heart axis. *Croat Med J*, 40(1):77–79, Mar 1999.

[19] B. Acar, G. Yi, and M. Malik. Concept of t-wave morphology dispersion. In *Computers in Cardiology 1999. Vol.26*, pages 57–60, 1999.

[20] M. O. Biagetti, P. D. Arini, E. R. Valverde, G. C. Bretran, and R. A. Quinteiro. Role of dipolar and nondipolar components of the t wave in determining the t wave residuum in an isolated rabbit heart model. *Journal of Cardiovascular Electrophysiology*, 15(3):356–363, 2004.

[21] F. H. Baglivo, P. D. Arini, J. P. Martínez, and P. Laguna. Analysis of t wave morphology parameters with signal averaging during ischemia induced by percutaneous transluminal coronary angioplasty. In *2009 36th Annual Computers in Cardiology Conference (CinC)*, pages 689–692, 2009.

[22] H. Wu, K. H. K. Patel, X. Li, B. Zhang, C. Galazis, N. Bajaj, A. Sau, X. Shi, L. Sun, Y. Tao, H. Al-Qaysi, L. Tarusan, N. Yasmin, N. Grewal, G. Kapoor, J. W. Waks, D. B. Kramer, N. S. Peters, and F. S. Ng. A fully-automated paper ECG digitisation algorithm using deep learning. *Sci Rep*, 12(1):20963, Dec 2022.