PulseLab: An Integrated and Expandable Toolbox for Pulse Wave Velocity-based Blood Pressure Estimation

Weinan Wang*, Pedram Mohseni†, Kevin Kilgore+ and Laleh Najafizadeh*

***Abstract*— In this paper, we introduce *PulseLab*, a comprehensive MATLAB toolbox that enables estimating the blood pressure (BP) from electrocardiogram (ECG) and photoplethysmogram (PPG) signals using pulse wave velocity (PWV)-based models. This universal framework consists of 6 sequential modules, covering end-to-end procedures that are needed for estimating BP from raw PPG/ECG data. These modules are "dataset formation", "signal pre-processing", "segmentation", "characteristic-points detection", "pulse transit time (PTT)/ pulse arrival time (PAT) calculation", and "model validation". The toolbox is expandable and its application programming interface (API) is built such that newly-derived PWV-BP models can be easily included. The toolbox also includes a user-friendly graphical user interface (GUI) offering visualization for step-by-step processing of physiological signals, position of characteristic points, PAT/PTT values, and the BP regression results. To the best of our knowledge, *PulseLab* is the first comprehensive toolbox that enables users to optimize their model by considering several factors along the process for obtaining the most accurate model for cuff-less BP estimation.**

I. Introduction

Cuff-less blood pressure (BP) estimation methods have been long pursued as substitutions for the conventional BP measurement methods, such as auscultation, oscillometry, volume clamping, and catheterization, to facilitate continuous and automatic monitoring of BP. The "model-driven" methods constitute one class of cuff-less BP estimation methods. Model-driven methods are developed based on the models that describe the relationship between the pulse wave velocity (PWV) and the BP. To date, several PWV-BP models have been derived and used in BP-estimation studies. Examples include the logarithmic model [1], the linear model [2], the inverse model [3], the inverse square model [4], and the mean BP model [5].

Several factors influence the BP estimation accuracy of the model-driven methods. For example, it has been shown that defining the characteristic points differently [6] or variability in extracting pulse transit time (PTT)/ pulse arrival time (PAT) [7] can affect the regression performance. It has also been shown [8] that the type of filters used in the pre-processing step of physiological signals could change the position of characteristic points, and hence, influencing the BP estimation accuracy. It is therefore important to be able to consider various options for pre-processing of physiological signals, definitions of characteristic points, and the choice of PWV-BP models, all in one place, to derive the most accurate BP-estimation model.

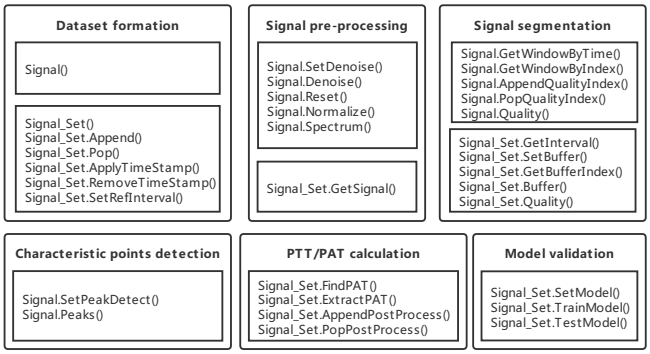


Fig. 1: Functions implemented in the API of *PulseLab*.

Currently, a number of toolboxes and graphical user interfaces (GUIs) exist that enable pre-processing, visualization and analysis of cardiovascular signals, but none targets the problem of BP estimation, and each has its own limitations for this application. For example, general signal processing toolboxes such as the MATLAB's built-in Signal Processing Toolbox provide the capabilities of filtering and segmentation to pre-process raw physiological signals, but they are not designed for further signal analysis required for BP estimation. Dedicated cardiovascular signal processing toolboxes such as the ecg-kit [9], the WFDB [10] or the Bio-SP [11] are capable of pre-processing and analyzing the quality of physiological signals such as electrocardiogram (ECG), photoplethysmogram (PPG) and impedance cardiogram (ICG), that are being frequently used in model-driven BP estimation studies, but are designed to analyze each signal individually and lack the capability of extracting PWV indicators, and doing regression analysis. Therefore, to the best of our knowledge, no existing biomedical signal processing toolbox has the full capability of supporting the needs for model-driven cuff-less BP estimation studies.

In this paper, we present *PulseLab*, a MATLAB toolbox, which offers a unified framework for end-to-end model-based BP estimation. The proposed toolbox consists of 6 modules: "dataset formation", "signal pre-processing",

This work was supported by Craig H. Neilsen Foundation Award # 598202.

* Department of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ 08854, USA. {ww329, laleh.najafizadeh}@rutgers.edu.

† Department of Electrical, Computer, and Systems Engineering, Case Western Reserve University, Cleveland, OH, USA. pxm89@case.edu.

+Departments of Orthopaedics and Physical Medicine & Rehabilitation, Case Western Reserve University and MetroHealth System, Cleveland, OH, USA. klk4@case.edu.

PulseLab is available for download at http://thepulselab.info.

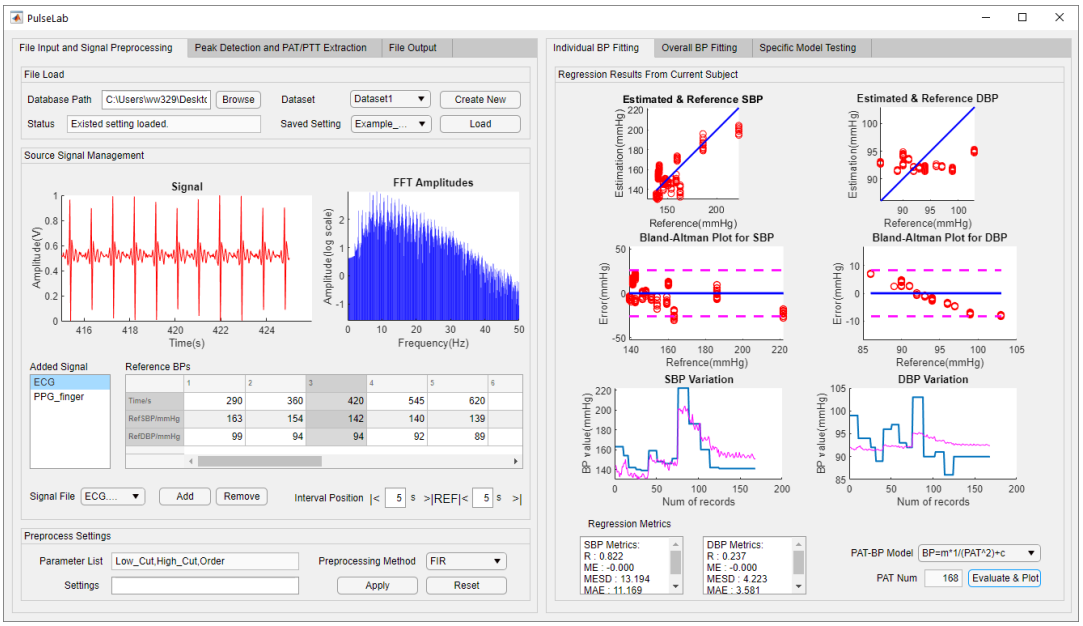


Fig. 2: Main appearance of the *PulseLab* GUI. The panels in the *File Input and Signal Preprocessing* tab and the *Individual BP Fitting* tab are displayed.

"segmentation", "characteristic points detection", "PTT/PAT calculation", and "model validation". The framework is implemented as an application programming interface (API), therefore, it can easily be expanded to include newly-derived PWV-BP models. The toolbox comes with a user-friendly GUI that provides visualization of the end-to-end BP estimation process, including plots of physiological waveforms, positions of characteristic points, extracted PAT/PTT values, and regression plots, which change dynamically after every parameter modification, offering feedback to user for better parameter tuning.

The rest of the paper is as follows. In Section II we describe the API and GUI of *PulseLab*. In Section III, we present testing examples of using this toolbox for BP-estimation based on real data. Finally the paper is concluded in Section IV.

II. Methods

A. Architecture of API in PulseLab

The 6 modules in the API of the *PulseLab* are shown in Fig. 1. The modules are implemented with functions belonging to two classes: the ***Signal*** class and the ***Signal_Set*** class. The use of decorator design pattern is maximized in every module to enable easy renewal of all key functions, to conform with the rapid evolving BP estimation algorithms. The ***Append*** and ***Set*** functions in Fig. 1 load exterior objects to the ***Signal*** class and ***Signal_Set*** class, which define the behavior of functions such as ***Signal.Denoise()*** or ***Signal_Set.TrainModel()*** relating to all key operations in signal pre-processing, signal quality evaluation, characteristic points detection, selection of PAT/PTT values (postprocessing), and formation of PWV-BP models. As such, new methods can be easily added and used with the API without making modification to the framework itself, which results in maximal code re-usability.

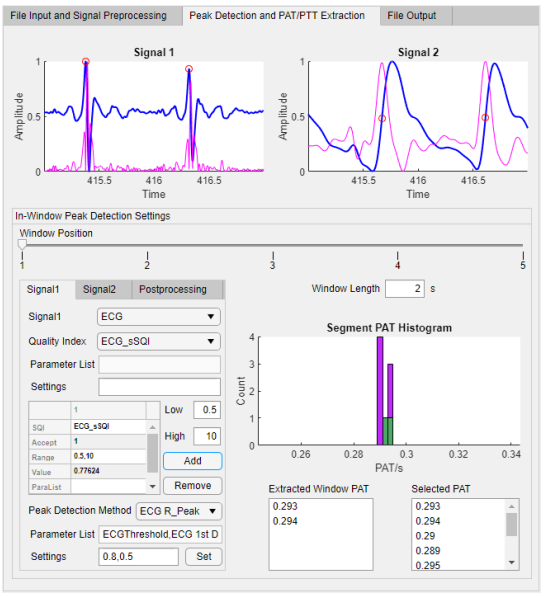
B. PulseLab GUI

The main appearance of the GUI and its different panels/options are shown in Fig. 2. The three tabs on the left side offer the operations for loading and pre-processing of raw physiological signals, and PAT/PTT extraction. The three tabs on the the right side conduct regression analysis between extracted PAT/PTT values and the reference BPs. In what follows, we describe some of these capabilities in more details.

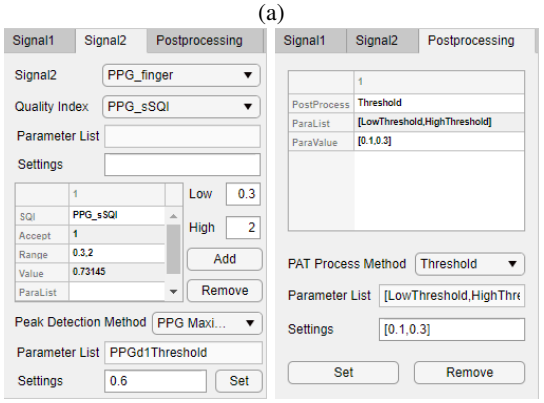
- The *File Input and Signal Processing* tab, offers the panels for loading the input raw ECG/PPG/reference BP data (in the .mat format), and visualization of the loaded waveforms (in time and frequency). The "Preprocess Settings" section in this panel offers various filter choices (such as FIR or IIR filters).

- The *Peak Detection and PAT/PTT Extraction* tab, offers the panels for segmenting signals and extracting the characteristic points/PAT/PTT from the signals. PAT/PTT are defined as the time difference between the characteristics points of the ECG and PPG signals within the same cardiac cycle. In the example shown in Fig. 3a, the synchronized ECG and PPG signals, both with 10-second duration, are segmented into five 2-second windows that include 1∼2 beat(s). One window, for ECG as "Signal1" (the proximal signal) and for PPG as "Signal2" (the distal signal), is displayed in Fig. 3a. By rolling the "Window Position" option, the user can inspect each cycle of the ECG/PPG signal to ensure proper PAT/PTT extraction.

The GUI also offers the options to exclude windows with poor signal quality by applying a threshold on the evaluated signal quality index (SQI). The GUI considers kurtosis (kSQI) and spectral SQI for the ECG signal [12], and skewness SQI (sSQI) for the PPG signal [13] as SQI. Examples of setting spectral SQI for an ECG recording and

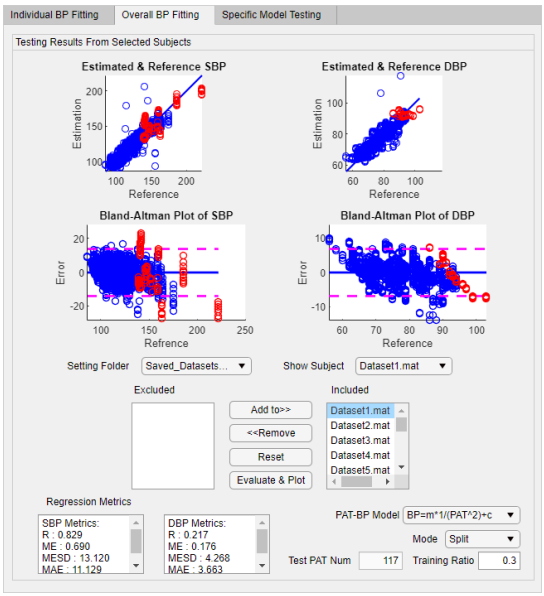


(a)

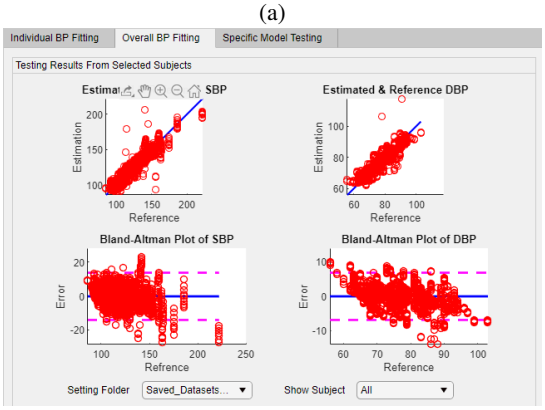


(b)

Fig. 3: Panels in the *Peak Detection and PAT/PTT Extraction* tab in the *PulseLab* GUI, offering options and settings for signal segmentation, detection of characteristic points, and PAT/PTT calculation. (a): The "Signal1" tab displays proximal signal settings, including the SQI and the detection of the characteristic points. (b): Left- The "Signal2" tab displays distal signal settings, including the SQI and the detection of characteristic points; Right- Settings for removing outliers from extracted PAT/PTT values (postprocessing).



(a)



(b)

Fig. 4: The panel in the *Overall BP Fitting* tab in the *PulseLab* GUI. (a): The regression plot and the Bland-Altman plot of Dataset1 (red circles) among all estimated and reference BP values in the whole database (blue circles). (b): The regression plot and Bland-Altman plot for all estimated and reference BP values in the whole database (red circles).

sSQI for a PPG recording are demonstrated in Fig. 3a and the left figure of Fig. 3b, respectively.

The panels in Fig. 3a and the left figure of Fig. 3b also list options conforming with different definitions of characteristic points, such as the R-peak of the ECG signal and the foot point, maximum slope and the systolic peak of the PPG signal. The example shows selecting R-peak for the ECG signal and the maximum slope for the PPG signal. The "Signal 1" and "Signal 2" figures in Fig. 3a change according to these settings, with blue curves indicating signal with acceptable quality, and red circles indicating the position of the characteristic points.

Once the characteristic points have been located for both proximal and distal signals, the values of PAT/PTT are automatically calculated for every window, and are listed under the "Extracted Window PAT" (see Fig. 3a). The values under the "Selected PAT" show the extracted PAT/PTT values for all windows. These values are also displayed as a histogram under the "Segment PAT Histogram", which can aid the user to remove PAT/PTT outliers with the post-processing settings listed in the right figure of Fig. 3b.

- The *File Output* tab (not shown), offers saving the settings and results for the current analysis to files. Previous settings can be restored from these saved files using the "File Load" panel displayed in Fig. 2.

- The *Individual BP Fitting* tab offers the panel for selecting a PWV-BP model and for fitting the extracted PAT/PTT values to the reference BP values (see Fig. 2, in which the inverse square model has been chosen). Regression metrics

TABLE I: Implementation of 5 testing examples using the *PulseLab* toolbox.

Implementation With PulseLab						Overall BP Estimation Performances			
Settings No.	***Signal1***	**Characteristic Point 1**	***Signal2***	**Characteristic Point 2**	**PWV-BP Model**	**R (SBP)**	**ME±SD (SBP)**	**R (DBP)**	**ME±SD (DBP)**
1	ECG	R-peak	fingertip PPG	foot point	Logarithmic	0.908	-0.512±8.793	0.904	-0.148±3.622
2	ECG	R-peak	fingertip PPG	foot point	Inverse	0.916	-0.008±8.203	0.913	0.078±3.448
3	ECG	R-peak	fingertip PPG	foot point	Inverse Square	0.920	-0.358±8.084	0.907	0.066±3.574
4	ECG	R-peak	fingertip PPG	maximum slope	Inverse Square	0.926	0.311±7.736	0.905	0.176±3.599
5	ECG	R-peak	fingertip PPG	systolic peak	Inverse Square	0.922	-0.180±7.836	0.906	0.015±3.551

including Pearson's correlation coefficients (R), mean error (ME) and mean absolute error (MAE) are calculated and reported for both the systolic blood pressure (SBP) and the diastolic blood pressure (DBP). The regression plot, the Bland-Altman plot and the BP variation plots (visualizing how estimated BPs track the variation in reference BPs) are also displayed in this panel. These regression results change dynamically if any of the parameters and settings in the tabs on the left side of GUI needs to be tuned or modified, providing real-time feedback to the user for optimizing the model.

- The *Overall BP Fitting* tab, offers the panel for performing general regression analysis across all subjects, in case there are multiple subjects in the database. PAT/PTT values extracted for each subject are saved within the *File Output* tab, and are loaded to this panel for overall analysis. The user can examine the performance of each dataset individually (Fig. 4a), or the overall performance by merging all reference and estimated BP values (Fig. 4b).

- The *Specific Model Testing* tab (not shown), offers the panel for training a model with some data, then testing the model on others. This can be useful for cases when one subject has multiple recordings in the database.

III. Results

To demonstrate the applicability of the proposed toolbox for BP estimation, we used data from 18 healthy subjects (aged from 25 to 58). ECG and fingertip PPG signals recorded at 10 kHz for each subject were downsampled to 1 kHz. The reference values for SBP and DBP were recorded using a cuff-based Omron 10 BP786N blood pressure monitor. During measurement, variations in BP values were created via physical exercises (running and stair climbing). Data was collected under an IRB-approved protocol at Metrohealth System.

We implemented five testing examples with the *PulseLab* toolbox and our data. The combinations of module settings and the BP estimation performances are summarized in Table I. As can be seen, for this dataset, the combination of extracting PAT with PPG maximum slope, and using the inverse square model to estimate BP from PAT, produces the best SBP prediction results, while using PPG foot point to extract PAT, and inverse model to estimate BP, produces the best DBP prediction results.

IV. Conclusion

In this paper, we introduced *PulseLab*, an end-to-end MATLAB toolbox for implementing model-driven cuff-less blood pressure estimation methods. The toolbox includes both API and GUI and offers graphical visualization at different steps of processing to aid with optimum parameter tuning. The toolbox can be easily expanded owing to the decorator design pattern in all key operations, which enables for the new functions, methods and algorithms to be added with no modification to the main framework of the API and GUI. The toolbox is capable of including various BP estimation algorithms. As such, the *PulseLab* toolbox is an integrated and expandable toolbox that is expected to be very useful for studies related to model-driven cuff-less BP estimation.

V. Acknowledgment

We thank Fatemeh Marefat and Reza Erfani, formerly with Case Western Reserve University, for obtaining the human subject data used in this study.

References

[1] J. Muehlsteff, *et al.*, "Cuffless estimation of systolic blood pressure for short effort bicycle tests: the prominent role of the pre-ejection period," in *Int. Conf. of the IEEE Engineering in Medicine and Biology Society*, 2006, pp. 5088–5092.

[2] W. Chen, *et al.*, "Continuous estimation of systolic blood pressure using the pulse arrival time and intermittent calibration," *Medical and Biological Engineering and Computing*, vol. 38, no. 5, pp. 569–574, 2000.

[3] M. Mase, *et al.*, "Feasibility of cuff-free measurement of systolic and diastolic arterial blood pressure," *Journal of Electrocardiology*, vol. 44, no. 2, pp. 201–207, 2011.

[4] P. Fung, *et al.*, "Continuous noninvasive blood pressure measurement by pulse transit time," in *Int. Conf. of the IEEE Engineering in Medicine and Biology Society*, vol. 1, 2004, pp. 738–741.

[5] C. Poon and Y. Zhang, "Cuff-less and noninvasive measurements of arterial blood pressure by pulse transit time," in *Int. Conf. of the IEEE Engineering in Medicine and Biology Society*, 2006, pp. 5877–5880.

[6] M. Kachuee, *et al.*, "Cuffless blood pressure estimation algorithms for continuous health-care monitoring," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 4, pp. 859–869, 2016.

[7] R. C. Block, *et al.*, "Conventional pulse transit times as markers of blood pressure changes in humans," *Scientific Reports*, vol. 10, no. 1, pp. 1–9, 2020.

[8] M. Elgendi, *et al.*, "The use of photoplethysmography for assessing hypertension," *NPJ Digital Medicine*, vol. 2, no. 1, pp. 1–11, 2019.

[9] A. Demski and M. L. Soria, "ecg-kit: a matlab toolbox for cardiovascular signal processing," *Journal of Open Research Software*, vol. 4, no. 1, 2016.

[10] I. Silva and G. B. Moody, "An open-source toolbox for analysing and processing physionet databases in matlab and octave," *Journal of Open Research Software*, vol. 2, no. 1, 2014.

[11] M. Nabian, *et al.*, "An open-source feature extraction tool for the analysis of peripheral physiological data," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 6, pp. 1–11, 2018.

[12] Q. Li, *et al.*, "Robust heart rate estimation from multiple asynchronous noisy sources using signal quality indices and a kalman filter," *Physiological Measurement*, vol. 29, no. 1, p. 15, 2007.

[13] M. Elgendi, "Optimal signal quality index for photoplethysmogram signals," *Bioengineering*, vol. 3, no. 4, p. 21, 2016.