Validation of a Python-Based Graphical User Interface for Calculation of Relative Fundamental Frequency[☆]

[*]Mara R. Kapsner-Smith, [†,‡]Aideen Gill, [*,§]Abigail Muzila, [*,§]Tamara Polyakova, [*,§]Austeja Rubeski, [†,‡]Joshua Kline, [*,§,¶]Cara E. Stepp, and [*,†,‡]Jenny Vojtech, *§¶*Boston, and* †‡*Natick, MA*

SUMMARY: Objectives. The purpose of this study was to validate PyRFF, a semi-automated software for calculating relative fundamental frequency (RFF). PyRFF was developed in Python, a widely used open source programming language, making it freely accessible and broadly applicable for researchers and clinicians. By providing an accessible tool for RFF extraction, PyRFF has the potential to facilitate wider adoption of RFF measures in both research and clinical settings.

Study Design. Validation study.

Methods. To assess the accuracy of PyRFF, RFF measures were extracted using both PyRFF and a previously validated, semi-automated algorithm implemented in MATLAB. The outputs from the two programs were then compared using statistical measures of error, including root mean square error (RMSE) and mean bias error (MBE).

Results. The results demonstrated near-identical outputs between PyRFF and the MATLAB-based algorithm. RMSE and MBE values were close to zero for all comparisons, indicating minimal discrepancy between the two methods.

Conclusions. PyRFF offers a valid, freely available software platform for extracting RFF measures, ensuring accessibility for a broad user base of researchers, clinicians, and students. The availability of free, user-friendly software is essential for advancing research and clinical application of RFF in voice assessment. By providing an alternative to MATLAB-based methods, PyRFF lowers financial and technical barriers, promoting further exploration of RFF as a potential tool in voice disorder assessment and treatment.

Key Words: Voice disorders–Relative fundamental frequency–Acoustic measures–Vocal hyperfunction–Voice–Dysphonia.

Abbreviations: GUI, graphical user interface–RFF, relative fundamental frequency–VfV, vowel-fricative-vowel–f_o, fundamental frequency.

INTRODUCTION

Vocal hyperfunction is characterized by muscle tension in and around the larynx during voicing and may occur as a primary cause of dysphonia or as a compensation for other conditions that impact the voice (eg, vocal fold paralysis;[1,2]). Clinical assessment of vocal hyperfunction relies heavily on subjective measures such as auditory-perceptual judgments of vocal effort, laryngeal palpation, and visual-perceptual judgments of videostroboscopy, which are prone to bias and poor reliability.[3-7] However, relative fundamental frequency (RFF) is an objective measure that has shown promise for clinical research and potentially for assessment of voice disorders, including hyperfunctional voice disorders, hyperkinetic dysarthria, and adductor-type laryngeal dystonia.[8-14] As a non-invasive and objective acoustic measure, RFF may provide an indirect measure of laryngeal tension.[14-16] Previous work has validated a semi-automated graphical user interface (GUI) for calculating RFF[17,18].[a] However, because the GUI was developed in MATLAB,[19] user accessibility is limited by the cost of software licenses and user comfort with running scripts in a development environment. A free software package that can be implemented independently of specific code development platforms, without requiring expensive software licenses or other secondary software, would facilitate broader validation and adoption of RFF measures.

RFF is a measure of cycle-to-cycle changes in the fundamental frequency (f_o) of vocal fold vibration in the last 10 cycles before (offset) and the first 10 cycles after (onset) phonation ceases for production of a voiceless consonant. In people with typical voices, f_o decreases slightly during offset cycles relative to steady-state phonation and increases at onset before gradually returning to steady-state levels.[14,20] RFF is thought to be affected by changes in vocal fold posture, laryngeal muscle tension, and transglottal pressure that occur during transitions between voiced and voiceless sounds.[16,20-22] RFF has been shown to modulate with emotional state in people with typical voices, likely due to change in muscular activation.[23] In

[a] Available at: https://sites.bu.edu/stepplab/research/rff/

Accepted for publication June 26, 2025.

[☆] This work was supported by the National Institute on Deafness and Other Communication Disorders under grant R01DC015570 (Stepp) and R43DC019585 (Vojtech), as well as by the De Luca Foundation.

From the *Department of Speech, Language, and Hearing Sciences, Boston University, Boston, MA 02215; †Delsys, Inc., Natick, MA 01760; ‡Altec, Inc., Natick, MA 01760; §Department of Biomedical Engineering, Boston University, Boston, MA 02215; and the ¶Department of Otolaryngology – Head and Neck Surgery, Boston University School of Medicine, Boston, MA 02215.

Address correspondence and reprint requests to: Mara Kapsner-Smith, Department of Speech, Language, and Hearing Sciences, Boston University, 111 Cummington Mall, Room 225 – 2nd floor, Boston, MA 02215. E-mail: mkapsner@bu.edu

Journal of Voice, Vol xx, No xx, pp. xxx–xxx
0892-1997

https://doi.org/10.1016/j.jvoice.2025.06.031

people with vocal hyperfunction, RFF measures may be atypical due, in part, to baseline elevated laryngeal tension.[10] Previous work has shown that RFF values are lower in speakers with voice disorders characterized by laryngeal tension, including vocal hyperfunction,[14,24] Parkinson's disease,[13] and laryngeal dystonia[8] compared to speakers without voice disorders. Measures derived from RFF, such as the values of the boundary cycles (offset cycle 10 and onset cycle 1), discriminate between speakers with and without hyperfunctional voice disorders and may have diagnostic utility.[10,11,25] Further, there is evidence that RFF measures are sensitive to voice therapy outcomes.[12,26] RFF also decreases in speakers without voice disorders in adverse speaking conditions such as wearing a face mask and in association with other markers of vocal fatigue,[27,28] and increases with training and implementation of vocal health strategies in the workplace.[29] Thus, RFF may provide a clinically meaningful indirect measure of laryngeal tension.

To date, clinical studies of RFF have been limited to controlled settings in a fairly small number of labs that have access to MATLAB software.[30] However, to fully establish the clinical utility of RFF, independent validation studies are needed across diverse clinical environments and populations. Expanding access to RFF analysis requires moving beyond MATLAB, which is costly and less accessible, to free alternatives. Although executable GUI-based MATLAB software may be produced using MATLAB Compiler to run outside the development environment, the resulting executables require installation of the MATLAB Runtime Environment (MCR). The MCR is a large (~1.5 GB) dependency that must be installed separately for each platform, and it must match the MATLAB version used for compilation. This creates two key barriers in clinical and educational settings: 1) Technical burden, as users must locate, install, and maintain a compatible runtime on their system; and 2) Feasibility due to file size and permissions. Hospital and clinic IT policies often restrict installation of large third-party runtimes, especially those that update frequently. Therefore, this study aimed to validate PyRFF, a Python-based tool for calculating RFF. Python is an open-source programming language that does not require costly software licenses. Python software can be packaged as lightweight, standalone executables that run across platforms without additional dependencies. By creating a user-friendly program that runs independently without specialized software, this study aimed to make RFF analysis more accessible to researchers and clinicians.

The development of PyRFF involved translating the freely available MATLAB-based "aRFF-AP" algorithm into Python. The algorithm was designed to extract RFF values from vowel–voiceless consonant–vowel utterances containing the fricative /f/ (ie, VfV). To calculate RFF, the algorithm comprises subroutines to (1) extract voice f_o and pitch strength contours for each audio file; (2) calculate probable fricative locations based on spectral energy content; (3) identify the last 10 cycles of phonation offset and the first 10 cycles of phonation onset using the f_o and pitch strength contours; and (4) calculate RFF measures, which are saved to an Excel spreadsheet.[31] Besides relying on MATLAB to run it, users had to manually manipulate parameters within a script, such as setting the minimum and maximum f_o values and specifying the absolute directory paths for audio input and RFF output folders. This required users to have a basic level of programming knowledge. In contrast, the updated version of PyRFF features a user-friendly GUI that prompts users to easily input values and select files or directories, eliminating the need to manually edit script parameters or type paths. This new interface streamlines the process, making it accessible to users without requiring prior knowledge of the MATLAB development environment or scripting.

Because implementing the algorithms used for semi-automated RFF calculation in Python required recreating core signal processing routines, validation against the original MATLAB-based implementation (aRFF-AP) was essential. Python and MATLAB use different signal processing libraries and computational backends, which can lead to subtle differences in outputs even when algorithms are theoretically identical. Thus, validating PyRFF using aRFF-AP as the gold standard ensured that the Python-based tool preserved the integrity of the original processing steps and produced comparable and reliable results. Using a test set of voice samples from individuals with and without voice disorders, we compared the outputs of PyRFF to those from the validated aRFF-AP tool. We hypothesized that the performance of the two programs would not exhibit clinically meaningful differences based on error comparisons between the Python and MATLAB algorithms in three areas: (1) automated f_o and pitch strength contour calculations, (2) consistency of automated fricative identification, and (3) RFF calculation. Validation of the Python-based RFF GUI (PyRFF) will improve accessibility of the measures, facilitating further research and potential clinical applications.

METHODS

Informed consent was obtained, and the study carried out in compliance with the Boston University Institutional Review Board (protocol #2625) or University of Washington Institutional Review Board (protocol #36181).

Participants

Extant speech recordings were selected from a database of study participants, comprising 100 speakers with a range of voice disorders and 100 control speakers with typical voices (see Table 1 for demographic information). The group with voice disorders required a diagnosis from a referring laryngologist. Voice diagnoses are listed in Table 2. The control group had no self-reported history of speech, language, or hearing disorders and were age- and sex-matched within a 5-year range.

For each participant, a speech-language pathologist—who was blinded to the study conditions—completed

TABLE 1.
Demographics and Perceptual Ratings of Voice Severity for Speakers Included in the Sample

Group	Age	Sex	Overall severity of dysphonia (0-100)
Control	*M* = 54 years Range 20-85 SD = 19.0	45 male 55 female	*M* = 13 Range 1-39 SD = 8.3
Voice Disorder	*M* = 54 years Range 18-84 SD = 19.1	45 male 55 female	*M* = 19 Range 0-100 SD = 19.6

TABLE 2.
Diagnosis Information for the Voice Disorder Group

Diagnosis	*n*
Parkinson's Disease	44
Primary Muscle Tension Dysphonia	35
Vocal Fold Nodules	7
Vocal Fold Edema	3
Vocal Fold Scarring	3
Upper Respiratory Infection	2
Vocal Fold Polyp	1
Vocal Fold Granuloma	1
Vocal Fold Paralysis	1
Gastroesophageal Reflux Disease	1

the 100-mm visual analog scale for overall severity of dysphonia from the Consensus Auditory-Perceptual Evaluation of Voice (CAPE-V) using the nonlinearly placed textual severity labels as originally published (American Speech-Language-Hearing Association, 2002). Mean overall severity of dysphonia scores were as follows: 13.0 (SD = 8.3, range = 1-39) for the control group and 19.0 (SD = 19.6, range = 0-100) for the voice disorder group.

Speech recordings

All signals were acquired digitally, and analysis occurred offline. Participants were recorded in a sound-attenuated booth, the waiting area of a clinic, or quiet room at one of three locations using Shure (Niles, IL, USA) microphones: (a) at Boston Medical Center with a dynamic headset microphone (model WH20XLR), (b) at Boston University with a condenser headset microphone (model SM35XLR), and (c) at the University of Washington with a dynamic headset microphone (model WH20XLR).

Acoustic signals were sampled at 44.1 kHz with 16-bit resolution. Speech recordings consisted of three tokens for each of three vowel–voiceless consonant–vowel combinations with the voiceless consonant /f/ (ie, VfV tokens): /ɑfɑ/, /ifi/, and /ufu/. Participants were instructed to produce the VfVs with equal stress on both syllables. Audio recordings were segmented into three.WAV files, one for each VfV type.

Procedures

Converting the RFF algorithm from MATLAB to Python

The MATLAB-based aRFF-AP algorithm described in Vojtech et al[18] was translated in Python 3.8.[32] See Lien et al[17] and Vojtech et al[18] for details regarding the development and refinement of the algorithm.

The structural organization of aRFF-AP was schematized to identify the individual scripts that comprise the algorithm. In total, the algorithm consisted of one main script entitled **main_RFF**, four nested scripts (**find_fo_aswipep**, **find_fric_aswipep**, **find_RFFcycles_aswipep**, **RFFoutput**), and 12 dependencies (ie, functions upon which the main or nested scripts rely to run; Figure 1).

Ensuring numerical equivalence across platforms required recreating core signal processing routines. MATLAB's built-in functions (eg, *filter*, *filtfilt*, *designfilt*, and *butter*) rely on proprietary backend optimizations and often include implicit scaling, padding, or normalization steps. These operations are not always transparent, which posed challenges in replicating identical behavior in Python. In contrast, Python's *scipy* and *numpy* libraries implement signal processing with stricter requirements for input formatting and without MATLAB's automatic handling of edge artifacts or default filter orderings.

For instance, converting the low-pass and high-pass filters used to process pitch strength and energy contours required explicit replication of phase and gain behavior by careful matching of MATLAB filter design parameters and zero-phase filtering behavior using *scipy.signal.filtfilt*. We encountered minor—but meaningful—differences in edge behavior that affected pitch strength estimation until manually corrected. Similarly, frequency-domain analyses required tuning Python's FFT operations to match MATLAB's default windowing conventions and resolution handling. These platform-specific differences meant that achieving numerical equivalence required deep reconstruction of signal processing components rather than direct code translation.

In addition, the dependencies used in aRFF-AP had to be translated and, in some cases, reimplemented in Python. Pitch estimation using Auditory-SWIPE' in MATLAB benefits from proprietary interpolation and error-reduction heuristics. In Python, we ported the open-source, MATLAB-based Auditory-SWIPE' from scratch using the `main_RFF`

Manages the overall RFF analysis workflow and allows users to define key parameters for signal processing such as input and output file paths, fo bounds, expected number of VfVs, and methods for displaying results

- `find_contours_aswipep`

  Estimates voice fo and pitch strength contours over time for the selected audio file using the Auditory-SWIPE' algorithm for fo estimation

  - `aswipep`

    Implements the Auditory-SWIPE' pitch estimation algorithm, which provides robust fo tracking based on sawtooth waveform matching principles

    - `ERBfilterbank`

      Creates an auditory-inspired filter bank based on equivalent rectangular bandwidth (ERB) scales used in the Auditory-SWIPE' algorithm for frequency analysis

    - `ERBfilters`

      Implements the individual ERB-scaled filters that decompose the audio signal into frequency bands for pitch analysis

    - `outmidear`

      Models outer and middle ear transfer characteristics to simulate auditory processing in the SWIPE algorithm

- `find_fric_aswipep`

  Automatically locates the probable positions of the voiceless consonant /f/ in VfV tokens using ratios of high- and low-frequency energy, with user confirmation or manual adjustment of locations

- `find_RFFcycles_aswipep`

  Uses identified fricative locations to determine 10 voicing cycles before voice offset (before the /f/) and 10 cycles after voice onset (after the /f/) using a sliding window approach based on estimated fo periods

  - `finer_t0_est`

    Provides refined timing estimates for cycle boundaries by analyzing local signal characteristics around initial cycle estimates

  - `trough_find`

    Identifies troughs in the acoustic signal to help delineate individual vocal cycles during phonation offset and onset

  - `peak_find`

    Identifies peaks in the acoustic signal to complement trough detection for precise vocal cycle boundary identification

  - `checking_cycles`

    Validates that identified vocal cycles meet quality criteria for periodicity and signal strength before including them in RFF calculations

  - `zero_crossings`

    Identifies zero-crossing points in the acoustic signal to assist with precise timing of cycle boundaries and phonation transitions

  - `threshfind`

    Determines appropriate amplitude thresholds for distinguishing voiced from voiceless segments based on local signal characteristics

  - `calcRFF`

    Computes the relative fundamental frequency values by calculating fo changes in semitones for offset cycles (relative to offset cycle 1) and onset cycles (relative to onset cycle 10)

- `RFFoutput`

  Uses cycle data to calculate final RFF measures or reject instances based on quality criteria, outputting results to an Excel spreadsheet with appropriate rejection codes and reasons

FIGURE 1. Structural organization of the aRFF-AP algorithm.

published specifications by Camacho,[33] aligning sampling windows, harmonic weights, and temporal smoothing explicitly. The translation of the algorithms was a collaborative, iterative process. The fricative detection process, which relies on spectral energy ratios, was also re-implemented using numpy.fft and band-specific filtering. Ensuring numerical stability and matching MATLAB's output required validation against edge cases and atypical fricative durations. Initially, the authors (A.M., T.P., and A.R.) converted the code line-by-line, followed by more comprehensive efforts (led by A.G. and J.V.) to ensure functional equivalence when transitioning from MATLAB's built-in functions to Python.

Initial validation used a "bottom-up" approach where individual Python functions were tested before integrating the complete workflow. To ensure consistent testing conditions, we generated reference variables by running aRFF-AP on a single /afa/ ("aafaa") recording in MATLAB, then used these same inputs to test the corresponding Python functions. This approach allowed systematic validation of each processing step before testing the complete algorithm.

After the 12 dependencies were converted, the five scripts involved in the overarching algorithm workflow were converted. The main_*RFF* script, which manages the overall process, allows users to define key parameters for signal processing (such as input and output file paths, f_o bounds, expected number of VfVs, and methods for displaying results). The Python version of this script was updated with prompts to simplify user input without requiring modifications to code. From here, there are three primary routines that prepare for RFF calculation. The first routine, *find_contours_aswipep*, estimates voice f_o and pitch strength over time for the selected file. This routine uses Auditory-SWIPE' for f_o estimation,[33] based on an evaluation of f_o estimators conducted by Vojtech et al.[18] The second routine, *find_fric_aswipep*, automatically locates the probable locations of the voiceless consonant /f/ in the VfV tokens within each sound file using ratios of high- and low-frequency energy. The user is then prompted to manually confirm or adjust the locations. The third, *find_RFFcycles_aswipep*, uses the fricative location(s) identified in the second routine, as well as manual corrections provided by the user via a GUI, to determine 10 voicing cycles before voice offset (ie, before the /f/) and 10 after voice onset (ie, after the /f/). Starting from the center of each fricative, a sliding window the length of the reciprocal of the f_o estimate (ie, the period) searches in both directions for peaks and troughs to identify vocal cycles in the offsets and onsets of the vowels (ie, boundary cycles). Finally, the *RFFoutput* routine uses the cycle data to calculate RFF or reject the instance based on criteria defined in Lien et al.[17] The f_o of each offset cycle is calculated in STs relative to offset cycle 1 and the f_o of each onset cycle is calculated relative to onset cycle 10. Offset cycle 1 and onset cycle 10 are considered "reference" cycles since they are closest to the midpoint of either vowel in a VfV token; in this way, they represent the f_o of (pseudo)steady-state phonation.

Updating the GUI architecture

Although aRFF-AP includes a GUI for viewing and correcting fricative locations (Figure 2A), the user was required to manually edit code to specify analysis parameters such as f_0 bounds and file paths. PyRFF integrates these functions into a comprehensive GUI that includes:

- Integrated file and folder selection dialogs with field validation for f_o bounds and other analysis parameters (Figure 2B).
- Navigation controls for reviewing previous analyses and resetting automatically detected fricative locations, with improvements based on user feedback (Figure 2C).
- Automated error logging with user-friendly messages.
- One-click export of results to Excel format.

These improvements eliminate the need for programming knowledge while maintaining the same analytical rigor as the original MATLAB implementation.

Evaluating the accuracy of RFF algorithm translation

After translating the entire RFF codebase from MATLAB to Python—resulting in the creation of the new RFF GUI called "PyRFF"—the accuracy of the translation was evaluated using the 200-speaker dataset. Audio files were processed with both aRFF-AP and PyRFF to extract voice onset and offset RFF measures. Three main evaluations were performed: (1) extraction of f_o and pitch strength contours via *find_contours_aswipep*, (2) automated identification of fricative locations using *find_fric_aswipep*, and (3) RFF calculations and/or token rejections (via *RFFoutput*) based on the same fricative locations as inputs to *find_RFFcycles_aswipep*.

To eliminate potential variability from user input during validation, a trained analyst (M.K.S.) manually identified and corrected fricative locations using the aRFF-AP interface. These corrected locations were then used as identical inputs for both MATLAB and Python RFF calculations, ensuring that any differences in output reflected algorithmic rather than user-input variations. The resulting RFF output for each VfV token included semitone (ST) values for voice offset cycles 1-10 (the last 10 cycles before the voiceless consonant) relative to offset cycle 1, and onset cycles 1-10 (the first 10 cycles after the voiceless consonant) relative to onset cycle 10. Any vowels that could not be analyzed were assigned a rejection code with a specific reason for failure, and these instances were compared between the two algorithm versions.

Statistical analysis

The f_o and pitch strength contours generated by *find_contours_aswipep* were compared across audio signals using

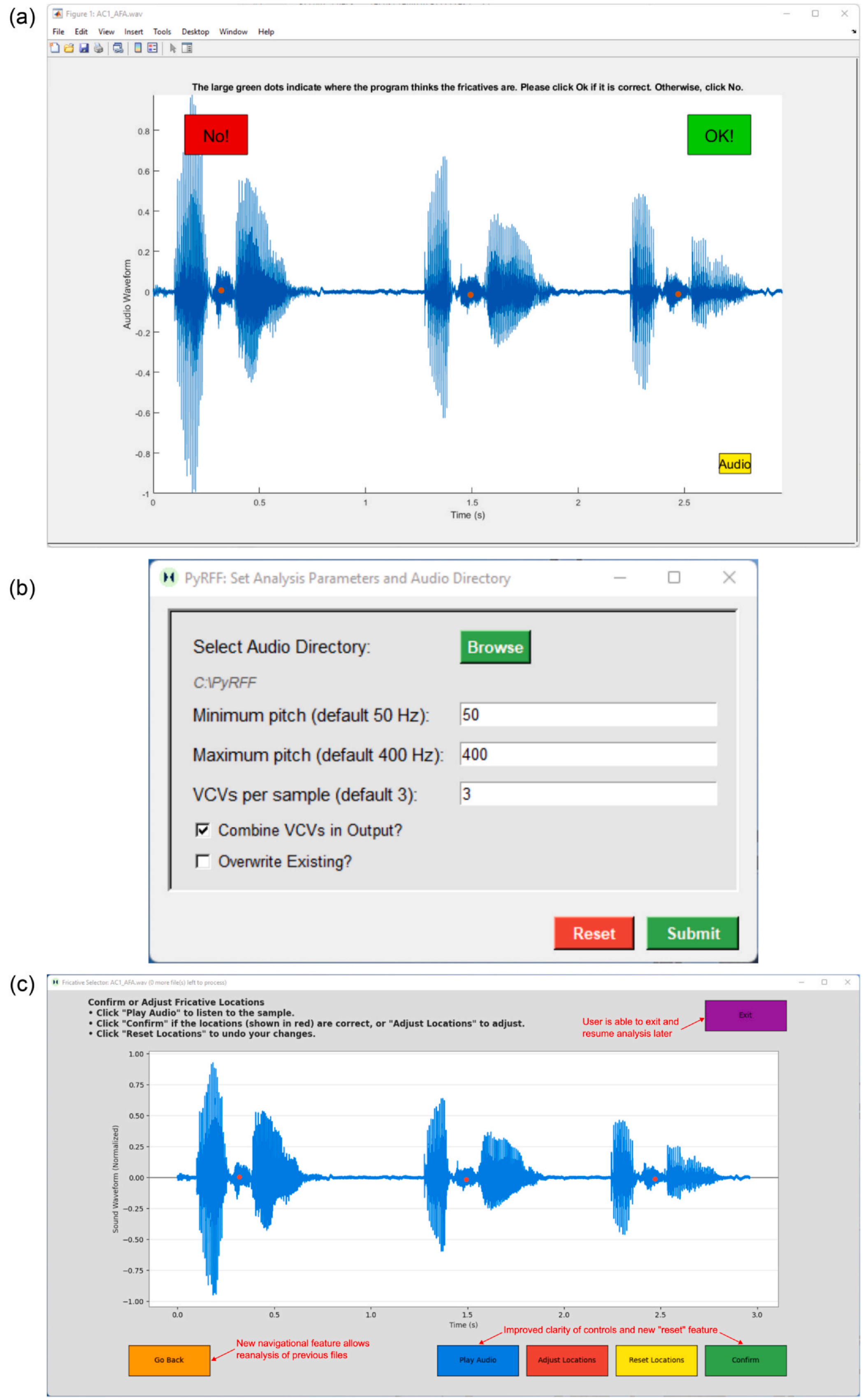

FIGURE 2. (**a**) Original aRFF-AP fricative selection GUI in MATLAB-based software. (**b**) Conversion of hard coded inputs to PyRFF GUI, including file locations and analysis parameters. (**c**) Improved PyRFF fricative selection GUI.

two error metrics: mean bias error (MBE) and root mean square error (RMSE). MBE estimated the average deviation between the contours from each algorithm, whereas RMSE measured the precision of the contours, regardless of whether the errors were positive or negative.

Automated fricative locations were compared by calculating the absolute time difference (in seconds) between the locations identified by the two algorithms. To normalize this comparison, the time difference was converted into a relative distance, expressed in terms of the average f_o measured in the audio file. This provided a measure of how many voice cycles separated the fricative locations.

For RFF values, results from the *find_RFFcycles_aswipep* and *RFFoutput* scripts were compared when using the same manually corrected fricative locations as inputs. The comparison focused on the mean values for voice offset cycles (1-10) and onset cycles (1-10), as well as for cycle 10 of voice offset and cycle 1 of voice onset. Previous research (eg, Heller Murray et al[10]) suggests that these particular cycles may be most sensitive to group differences in some clinical populations, which is why they were prioritized.

To assess the concurrent validity of PyRFF compared to the validated aRFF-AP algorithm, we calculated Pearson's r, RMSE, and MBE for the extracted RFF values from both programs. These statistical measures were chosen to evaluate the strength and accuracy of the relationship between the two algorithms' outputs, to confirm that PyRFF performs similarly to the original aRFF-AP algorithm.

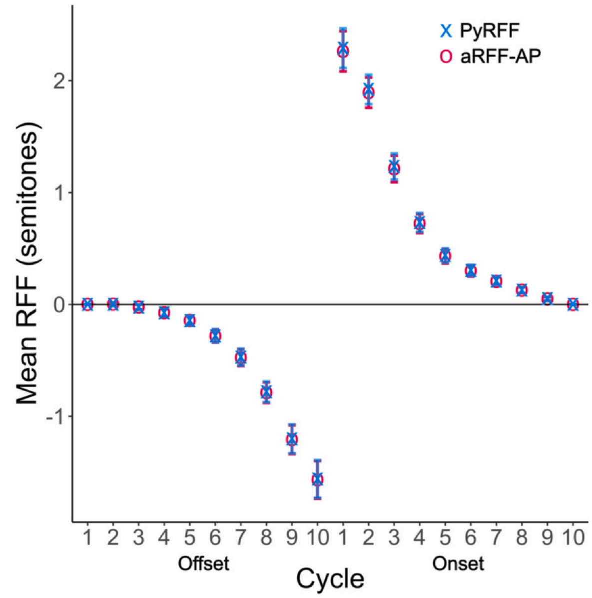

FIGURE 3. Mean RFF cycle values and 95% confidence intervals from aRFF-AP and PyRFF output.

RESULTS

For descriptive purposes, RFF cycle averages are presented in Figure 3 for each program. The group means and confidence intervals were nearly identical for the two programs. The type and number of RFF cycle rejections for each program are presented in Table 3. The cycle rejection results were nearly identical.

Contour extraction and fricative locations

Results of comparisons of f_o and pitch strength contours extracted automatically in the MATLAB software and PyRFF are shown in Table 4. The mean RMSE and mean MBE were near zero for both comparisons. The mean difference in automated (ie, not manually verified or corrected) fricative locations between the two programs was < 0.01 cycles (SD = 0.01). All fricative locations were verified manually. Of 600 audio files, 129 required manual adjustments of one or more fricative locations.

RFF calculations

Results for comparison of RFF calculations between the MATLAB software and PyRFF are shown in Table 5. All RMSE and MBE values were close to zero, and Pearson's r was 1.00 for all comparisons, suggesting near-perfect correspondence between the output of the two programs.

To assess the robustness of RFF analysis across different voice qualities, we compared rejection rates between speaker groups as part of a post hoc analysis. Higher rejection rates in dysphonic voices would indicate potential limitations for clinical application, whereas similar rates would suggest robust performance across voice disorders. Results are shown in Table 6. Overall, there were approximately 7% more rejections in the voice disorder group than the control group. There were two speakers in the voice disorder group with no analyzable RFF cycles. All control speakers had analyzable RFF cycles.

DISCUSSION

RFF calculations were extracted using a software package written in Python[32] and were compared to output from a previously validated semi-automated algorithm[17,18] written in MATLAB.[19] Results were compared for absolute errors (MBE, RMSE) as well as consistency of RFF values (Pearson's r). The two programs performed similarly, with very small average errors in f_o and pitch strength contour extraction (mean RMSE = 0.09 and < 0.01, respectively; mean MBE < 0.01), and very small average differences in automated fricative location (< 0.01 cycles). After manual adjustments of fricative locations, RMSE and MBE statistics calculated on RFF calculations indicated near zero error between the two programs, with perfect correlation. Although the two programs perform the same data extraction and calculation procedures, differences can occur across platforms (eg, MATLAB vs. Python), for example, due to differences in how functions are written, such as filters used for audio processing, or small differences in how floating point and complex numbers are handled in the language. Thus, this study was necessary to ensure that

TABLE 3.
RFF Cycle Rejection Totals for aRFF-AP and PyRFF

	Offset cycles		Onset cycles	
Rejection reason	aRFF-AP	PyRFF	aRFF-AP	PyRFF
Too few periodic cycles	195	195	406	407
Fails to reach steady state or unstable	0	0	0	0
Glottalized	19	19	69	69
Aperiodic or irregular	0	0	0	0
Sharp transition	657	657	355	354

TABLE 4.
Mean Errors in Contours Extracted Automatically in the Matlab and Python Programs, Averaged Across Speakers

Contour	Mean RMSE	Mean MBE
f_o (Hz)	0.09 (SD = 0.43)	7.17E-4 (SD = 0.01)
Pitch strength (a.u.)	−1.79E-6 (SD = 2.22E-6)	9.34E-6 (SD = 1.62E-5)

TABLE 5.
Comparisons of RFF Output From Semi-automated MATLAB and Python Programs With Manual Corrections Made in MATLAB and Imported Into Python

RFF Variable	*r*	RMSE	MBE
Overall Mean	1.00 (P < 0.001)	1.40E-9	9.24E-11
Mean Offset	1.00 (P < 0.001)	1.32E-9	−8.53E-12
Mean Onset	1.00 (P < 0.001)	1.36E-9	4.20E-11
Offset 10	1.00 (P < 0.001)	1.53E-9	1.23E-10
Onset 1	1.00 (P < 0.001)	1.39E-9	−1.28E-10

TABLE 6.
RFF Cycle Rejection Totals by Speaker Group

Rejection reason	Control	Voice disorders
Too few periodic cycles	263	339
Fails to reach steady state or unstable	0	0
Glottalized	46	42
Aperiodic or irregular	0	0
Sharp transition	513	498
Total rejections	822	879

differences were minimized and non-significant between the two implementations of RFF.

One challenge in the acoustic analysis of dysphonic voices is the robustness of a given measure to noisy or aperiodic signals.[34] As a post hoc analysis, the number and type of RFF cycle rejections were compared between speakers with and without voice disorders. There were 7% more RFF cycle rejections in the voice disorder group than the control group, with the majority of these rejections being due to the algorithm being unable to locate periodic voice cycles during signal preprocessing (*find_fric_aswipep*). This resulted in 2/100 speakers in the voice disorder group with no analyzable RFF cycles, compared to 0/100 in the control group. These results reflect a relatively high success rate for acoustic analysis of dysphonic voices, with only 2% of the voice disorder sample that did not have any usable cycles.

Although these findings suggest that RFF may be a fairly robust acoustic measure for these signal types, it is important to consider the perceived severity of dysphonia in the analyzed sample. The MATLAB-based algorithm was enhanced in Vojtech et al[18] through rule-based signal processing incorporating the measure of pitch strength[35] to account for differences in dysphonia severity. However, the algorithm may still struggle with more aperiodic voices. In the voice disorder group, severity spanned the full scale (0-100), whereas in the control group, it ranged only from 1 to 39. Although this distinction is important, the mean dysphonia severity values were relatively similar between groups (19 vs. 13), suggesting that, on average, the voice disorder group did not exhibit severe dysphonia—an important consideration when interpreting these findings.

Development of a standalone Python executable for semi-automated RFF calculation has several technical advantages over MATLAB. Deployment of executable software compiled in MATLAB requires that the user install a MATLAB runtime environment (MCR), a large dependency that must be installed separately for each platform and must match the MATLAB version used for compilation. These technical barriers may be prohibitive in clinical and education settings due to IT policies and user knowledge requirements. Further, it may complicate deployment of updated versions of software compiled in newer versions of MATLAB. In contrast, PyRFF was implemented in pure Python and compiled into a small, cross-platform standalone executable. It requires no runtime environment or administrator permissions. This makes PyRFF substantially more accessible for clinicians, students, and researchers unfamiliar with software development.

It is important to note that the scope of this paper is limited to concurrent validation of PyRFF relative to the previously validated MATLAB semi-automated RFF algorithm.[18] This concurrent validation provides no information about the validity or interpretation of RFF measures themselves or the reliability of user input. Information about the validity, interpretation, and potential clinical utility of RFF measures has been published in a number of previous studies (for a review related to voice disorders, see McKenna et al[30]). Future studies are recommended to assess usability of the enhanced interface and optimization benchmarking.

Our findings support the concurrent validity of PyRFF compared to a previously validated semi-automated GUI for RFF extraction.[17,18] PyRFF is freely available for download at https://sites.bu.edu/stepplab/research/rff/ and requires no experience with coding or development environments to install and use. The availability of free and user-friendly software to extract RFF is a crucial step for clinical validation and application. Further research—facilitated by the availability of this software—may lead to an affordable, accessible, and non-invasive clinical tool for assessment of vocal hyperfunction.

CONCLUSIONS

Although RFF has shown promise as an objective acoustic indicator of vocal hyperfunction, the accessibility of the measure was previously hindered by high-cost software licensing and the need for basic programming skills. This study provides evidence for the concurrent validity of PyRFF, a freely available software package for RFF extraction based on the previously validated semi-automated MATLAB algorithm, aRFF-AP. PyRFF offers a user-friendly, cost-free solution for measuring RFF to facilitate downstream clinical validation and implementation.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships, which may be considered as potential competing interests: Cara Stepp reports financial support was provided by the National Institute on Deafness and Other Communication Disorders. Jenny Vojtech reports financial support was provided by the National Institute on Deafness and Other Communication Disorders. Aideen Gill reports financial support was provided by the National Institute on Deafness and Other Communication Disorders. Joshua Kline reports financial support was provided by the National Institute on Deafness and Other Communication Disorders. Jenny Vojtech reports financial support was provided by the De Luca Foundation. Aideen Gill reports financial support was provided by the De Luca Foundation. Joshua Kline reports financial support was provided by the De Luca Foundation. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Belafsky PC, Postma GN, Reulbach TR, et al. Muscle tension dysphonia as a sign of underlying glottal insufficiency. *Otolaryngol–Head Neck Surg.* 2002;127:448–451. https://doi.org/10.1067/mhn.2002.128894.
2. Hillman RE, Stepp CE, Van Stan JH, et al. An updated theoretical framework for vocal hyperfunction. *Am J Speech-Lang Pathol.* 2020;29:2254–2260. https://doi.org/10.1044/2020_AJSLP-20-00104.
3. Eadie TL, Sroka A, Wright DR, Merati A. Does knowledge of medical diagnosis bias auditory-perceptual judgments of dysphonia? *J Voice.* 2011;25:420–429. https://doi.org/10.1016/j.jvoice.2009.12.009.
4. Sauder C, Nevdahl M, Kapsner-Smith M, et al. Does the accuracy of case history affect interpretation of videolaryngostroboscopic exams? *Laryngoscope.* 2020;130:718–725.
5. Stepp CE, Heaton JT, Braden MN, et al. Comparison of neck tension palpation rating systems with surface electromyographic and acoustic measures in vocal hyperfunction. *J Voice Off J Voice Found.* 2011;25:67–75. https://doi.org/10.1016/j.jvoice.2009.08.001.
6. Yiu EM, Lau VC, Ma EP, et al. Reliability of laryngostroboscopic evaluation on lesion size and glottal configuration: A revisit. *Laryngoscope.* 2014;124:1638–1644. https://doi.org/10.1002/lary.24521.
7. Zraick R, Kempster GB, Connor NP, et al. Establishing validity of the consensus auditory-perceptual evaluation of voice (CAPE-V). *Am J Speech-Lang Pathol.* 2011;20:14–22. https://doi.org/10.1044/1058-0360(2010/09-0105).
8. Eadie TL, Stepp CE. Acoustic correlate of vocal effort in spasmodic dysphonia. *Ann Otol Rhinol Laryngol.* 2013;122:169–176. https://doi.org/10.1177/000348941312200305.
9. Goberman AE, Blomgren M. Fundamental frequency change during offset and onset of voicing in individuals with Parkinson disease. *J Voice.* 2008;22:178–191. https://doi.org/10.1016/j.jvoice.2006.07.006.
10. Heller Murray ES, Lien YS, Van Stan JH, et al. Relative fundamental frequency distinguishes between phonotraumatic and non-phonotraumatic vocal hyperfunction. *J Speech Lang Hear Res JSLHR.* 2017;60:1507–1515.
11. Kapsner-Smith MR, Díaz-Cádiz ME, Vojtech JM, et al. Clinical cutoff scores for acoustic indices of vocal hyperfunction that combine relative fundamental frequency and cepstral peak prominence. *J Speech Lang Hear Res.* 2022;65:1349–1369.
12. Roy N, Fetrow RA, Merrill RM, Dromey C. Exploring the clinical utility of relative fundamental frequency as an objective measure of vocal hyperfunction. *J Speech Lang Hear Res JSLHR.* 2016;59:1002–1017. https://doi.org/10.1044/2016_jslhr-s-15-0354.
13. Stepp CE. Relative fundamental frequency during vocal onset and offset in older speakers with and without Parkinson's disease. *J Acoust Soc Am.* 2013;133:1637–1643.
14. Stepp CE, Hillman RE, Heaton JT. The impact of vocal hyperfunction on relative fundamental frequency during voicing offset and onset. *J Speech Lang Hear Res.* 2010;53:1220–1226.
15. McKenna VS, Heller Murray ES, Lien Y-AS, Stepp CE. The relationship between relative fundamental frequency and a kinematic estimate of laryngeal stiffness in healthy adults. *J Speech Lang Hear Res JSLHR.* 2016;59:1283–1294. https://doi.org/10.1044/2016_JSLHR-S-15-0406.
16. Serry MA, Stepp CE, Peterson SD. Physics of phonation offset: towards understanding relative fundamental frequency observations. *J Acoust Soc Am.* 2021;149:3654–3664.
17. Lien YS, Heller Murray ES, Calabrese CR, et al. Validation of an algorithm for semi-automated estimation of voice relative fundamental frequency. *Ann Otol Rhinol Laryngol.* 2017;126:712–716. https://doi.org/10.1177/0003489417728088.
18. Vojtech JM, Segina RK, Buckley DP, et al. Refining algorithmic estimation of relative fundamental frequency: accounting for sample characteristics and fundamental frequency estimation method. *J Acoust Soc Am.* 2019;146:3184. https://doi.org/10.1121/1.5131025.
19. MathWorks. 2021. MATLAB 2021 [Computer software].
20. Watson BC. Fundamental frequency during phonetically governed devoicing in normal young and aged speakers. *J Acoust Soc Am.* 1998;103:3642–3647.
21. Ladefoged P. *Three Areas of Experimental Phonetics: Stress and Respiratory Activity, the Nature of Vowel Quality, Units in the*

Perception and Production of Speech. Oxford, U.K.: Oxford University Press,; 1967.
22. Stevens KN. Physics of laryngeal behavior and larynx modes. *Phonetica*. 1977;34:264–279. https://doi.org/10.1159/000259885.
23. van Mersbergen M, Lanza E. Modulation of relative fundamental frequency during transient emotional states. *J Voice*. 2019;33: 894–899. https://doi.org/10.1016/j.jvoice.2018.07.020.
24. Ferrán S, Rodríguez-Zanetti C, Garaycochea O, et al. Relative fundamental frequency: only for hyperfunctional voices? A pilot study. *Bioengineering*. 2024;11:475. https://doi.org/10.3390/bioengineering11050475.
25. Cheema AJ, Marks KL, Ghasemzadeh H, et al. Characterizing vocal hyperfunction using ecological momentary assessment of relative fundamental frequency. *J Voice*. 2024. https://doi.org/10.1016/j.jvoice.2024.10.025.
26. Stepp CE, Merchant GR, Heaton JT, Hillman RE. Effects of voice therapy on relative fundamental frequency during voicing offset and onset in patients with vocal hyperfunction. *J Speech Lang Hear Res*. 2011;54:1260–1266. https://doi.org/10.1044/1092-4388(2011/10-0274).
27. McKenna VS, Kendall CL, Patel TH, et al. Impact of face masks on speech acoustics and vocal effort in healthcare professionals. *Laryngoscope*. 2022;132:391–397. https://doi.org/10.1002/lary.29763.
28. McKenna VS, Patel TH, Kendall CL, et al. Voice acoustics and vocal effort in mask-wearing healthcare professionals: a comparison pre- and post-workday. *J Voice*. 2023;37:802.e15–802.e23. https://doi.org/10.1016/j.jvoice.2021.04.016.
29. McKenna VS, Gustin RL, Howell RJ, et al. Developing educational health modules to improve vocal wellness in mask-wearing occupational voice users. *J Voice*. 2024;38:966.e9–966.e18. https://doi.org/10.1016/j.jvoice.2021.11.006.
30. McKenna VS, Vojtech JM, Previtera M, et al. A scoping literature review of relative fundamental frequency (RFF) in individuals with and without voice disorders. *Appl Sci*. 2022;12:8121. https://doi.org/10.3390/app12168121.Article 16.
31. Microsoft Corporation. *Excel (Version Microsoft 365) [Computer software]*. Redmond, WA: Microsoft Corporation,; 2024.
32. Python Software Foundation. 2019. Python Language (Version 3.8) [Computer software].
33. Camacho A. *SWIPE: A Sawtooth Waveform Inspired Pitch Estimator for Speech and Music*. Gainesville, FL: University of Florida,; 2007.
34. Titze IR. *Workshop on Acoustic Voice Analysis: Summary Statement*. Denver, CO: National Center for Voice and Speech,; 1995.
35. Camacho A, Harris JG. A sawtooth waveform inspired pitch estimator for speech and music. *J Acoust Soc Am*. 2008;124:1638–1652.