YAAPT Pitch Tracking MATLAB Function

# INTRODUCTION

The YAAPT (Yet Another Algorithm for Pitch Tracking) is a fundamental frequency (Pitch) tracking algorithm, which is designed to be highly accurate and very robust for both high quality and telephone speech.

 The YAAPT program is developed by the Speech Communication Laboratory at the State University of New York (SUNY) at Binghamton. The entire algorithm is available at <http://www.ws.binghamton.edu/zahorian> as MATLAB functions. Further information including algorithm overview, implementation details, parameter settings and performance comparison can be found at S. A. Zahorian and H. Hu "A spectral/temporal method for robust fundamental frequency tracking," J.Acosut.Soc.Am. 123(6), June 2008.

To cite YAAPT in your publications, please refer to:

A spectral/temporal method for robust fundamental frequency tracking
Stephen A. Zahorian, Hongbing Hu
The Journal of the Acoustical Society of America, 123, 4559-4571

# CONTENTS

The YAAPT pitch tracking algorithm is implemented as a MATLAB function yaapt(), which checks input parameters and invokes a number of associated routines for the YAAPT pitch tracking. The function call format is described as follows:

 **[Pitch, numfrms, frmrate] = yaapt(Data, Fs, VU, ExtrPrm, fig, speed)**

 INPUTS:

 **Data:** Input speech acoustic samples

 **Fs:** Sampling rate of the input data

 **VU:** Whether to make voiced/unvoiced decisions with 1 for True and 0 for False.

 The default is 1. As 0, the entire acoustic segment is considered all voiced

**ExtrPrm:** Extra parameters in a struct type for performance control. Default values are given in a later section in a table. Some examples are given of details are given here.

 ExtrPrm.f0\_min = 60; % Change minimum search F0 to 60Hz

 ExtrmPrm.fft\_length = 8192; % Change FFT length to 8192

 **fig**: Whether to plot pitch tracks, spectrum, energy, etc. The parameter

 is 1 for True and 0 for False. The default is 0.

 **Speed:** choose processing speed of the YAPPT. This version of YAPPT

 has three processing speeds. 1 means the complete version

 of YAPPT, which has highest over all accuracy but slowest processing speed

 2 is a version only use linear signal in time domain

 processing, which has a balanced performance in both

 accuracy and processing speed. 3 is the spectral only

 version YAPPT, which only use frequency domain processing.

 This version is the fastest, but has lower accuracy. The

 default is 2.

 OUTPUTS:

 **Pitch:** Final pitch track in Hz. Unvoiced frames are assigned to 0s.

 **numfrms:**  Total number of calculated frames, or the length of output pitch track

 **frmrate:**  Frame rate of output pitch track in ms

##  Algorithm Control Parameters

The YAAPT utilizes a set of parameters for algorithm control, such as frame length, FFT length, and dynamic programming weights. By choosing different parameters, the YAAPT can be tuned to compute a pitch track with voiced/unvoiced decision for the minimum big error, or a track without voiced/unvoiced decision for the minimum gross error. The gross error is computed as the percentage of voiced frames, such that the pitch estimate of the tracker signiﬁcantly deviates (20% is generally used) from the pitch estimate of the reference, while the big error is equal to the number of voiced frames with gross errors plus the number of unvoiced frames erroneously labeled as voiced frames divided by the total number of frames [1]. When tracking without voicing decisions, YAAPT will first track to minimize big error and then apply some corrective measures explained in section 2.2.

The parameters and their optimal values for both the minimum big and gross errors as listed in Table 1 in the work of S. A. Zahorian and H. Hu [1]. In the program, the corresponding parameters are declared in a *struct* type. As shown below, Prm\_VU contains the default values for the tracking with voiced/unvoiced decision (big error) and Prm\_aV for the tracking with all frames voiced (gross error).

|  |
| --- |
| % Default values for the tracking with voiced/unvoiced decisionPrm\_VU = struct(...'frame\_length', 35, ... % Length of each analysis frame (ms) 'frame\_lengtht', 35, ... % Length of each time domain analysis  'frame\_space', 10, ... % Spacing between analysis frame (ms) 'f0\_min', 60, ... % Minimum F0 searched (Hz) 'f0\_max', 400, ... % Maximum F0 searched (Hz) 'fft\_length', 2048, ... % FFT length 'bp\_forder', 150, ... % Order of bandpass filter 'bp\_low', 50, ... % Low frequency of filter passband (Hz) 'bp\_high', 1500, ... % High frequency of filter passband (Hz) 'nlfer\_thresh1',0.75, ... % NLFER boundary for voiced/unvoiced decisions 'nlfer\_thresh2', 0.1, ... % Threshold for NLFER definitely unvoiced 'shc\_numharms', 3, ... % Number of harmonics in SHC calculation 'shc\_window', 40, ... % SHC window length (Hz) 'shc\_maxpeaks', 4, ... % Maximum number of SHC peaks to be found 'shc\_pwidth', 50, ... % Window width in SHC peak picking (Hz) 'shc\_thresh1', 5.0, ... % Threshold 1 for SHC peak picking 'shc\_thresh2', 1.25, ... % Threshold 2 for SHC peak picking 'f0\_double', 150, ... % F0 doubling decision threshold (Hz) 'f0\_half', 150, ... % F0 halving decision threshold (Hz) 'dp5\_k1', 11, ... % Weight used in dynamic program 'dec\_factor', 1, ... % Factor for signal resampling 'nccf\_thresh1', 0.3, ... % Threshold for considering a peak in NCCF 'nccf\_thresh2', 0.9, ... % Threshold for terminating search in NCCF 'nccf\_maxcands', 3, ... % Maximum number of candidates found 'nccf\_pwidth', 5, ... % Window width in NCCF peak picking 'merit\_boost', 0.20, ... % Boost merit 'merit\_pivot', 0.99, ... % Merit assigned to unvoiced candidates in ... % definitely unvoiced frames 'merit\_extra', 0.4, ... % Merit assigned to extra candidates ... % in reducing F0 doubling/halving errors 'median\_value', 7, ... % Order of medial filter 'dp\_w1', 0.15, ... % DP weight factor for V-V transitions 'dp\_w2', 0.5, ... % DP weight factor for V-UV or UV-V transitions 'dp\_w3', 0.1, ... % DP weight factor of UV-UV transitions 'dp\_w4', 0.9, ... % Weight factor for local costs 'end', -1); |

A convenient way to modify the above parameters is to use the input **ExtrPrm** parameter to yaapt(). For instance, assign desired parameter values using a struct **ExtrPrm** in the MATLAB command window**,** and pass ExtrPrm to the yaapt() function as follows,

|  |
| --- |
| >> ExtrPrm.f0\_min = 60; % Change minimum F0 searched to 60Hz>> ExtrPrm.f0\_max = 400; % Change maximum F0 searched to 400Hz>> ExtrPrm.fft\_length = 8192; % Change FFT length to 8192>> [pitch] = yaapt(data, fs, ExtrPrm); % Execute YAAPT with the changes |

##  Pitch Track Corrections

YAAPT includes an optional tool for applying some minor correction techniques to the calculated pitch track. This tool is contained in a MATLAB function called ptch\_fix(). This tool is designed to smooth the pitch track with interpolation for applications where an all voiced track is desired and includes options for correcting pitch halves or doubles and extrapolation.

 The tool is automatically applied when tracking for minimum gross error without voicing decisions. Therefore, it is typically not necessary to independently call this function. However, the tool will use a set of optimized default parameters unless directed otherwise. If alternate parameters are desired, the default parameter file called ptch\_fix.dat can be modified to reflect the desired parameters. The defaults are given in the following table.

|  |
| --- |
| PITCH\_HALF: 1 // attempt to eliminate pitch halves // 0 = no, 1 = zero them, 2 = double themPITCH\_HALF\_SENS: 2.8 // Sensitivity - number of std devs away // from the mean before being identifiedPITCH\_DOUBLE: 1 // Attempt to correct pitch doubles // 0 = no, 1 = zero them, 2 = halve themPITCH\_DOUBLE\_SENS: 3.3 // Sensitivity - number of std devs away  // from the mean before being identifiedSMOOTH\_INTERP: 1 // Smoothly interpolate unvoiced regions // 0 no, 1 yesSMOOTH\_FACTOR: 5 // Parameter for smoothing median filter // Odd numbers are best (default = 5)SMOOTH: 0 // Smooth the track (after interpolation) // 0=Disabled, # = median filter orderEXTRAP: 1 // Extrapolate: Set all leading and // trailing zeros to mean value |

To use the tool independently, call the function and pass a computed pitch track, such as is output from the yaapt function. The function call format is as follows:

**[Pitch\_out] = ptch\_fix(Pitch\_in,CmdFile)**

 INPUTS:

 **Pitch\_in:** Input pitch track values

 **CmdFile:** Input file with function parameters (Default: ptch\_fix.dat)

 OUTPUTS

 **Pitch\_out:** Pitch track after corrections

## Processing speed choose

 This version of YAPPT has three processing speeds, which can be controlled in the input parameter as 1, 2, 3.

**Speed 1** means the complete version of YAPPT, which has highest overall accuracy but slowest processing **speed 2** is a version only use linear signal in time domain processing, which has a balanced performance in both accuracy and processing speed. For this version, the processing speed improved 40% compared to the original version. While the accuracy only drops about 0.5% in studio speech with white noise case and clean simulated telephone case.

**Speed 3** is the spectral only version YAPPT, which only use frequency domain processing. This version is the fastest, but has lower accuracy. For this version, the processing speed improved about 79% compared to the original version, while the error rate only increased by about 1%.

# EXAMPLES

A number of examples are provided here to demonstrate how to use the YAAPT program in MATLAB for different scenarios. Two sample speech files (f1nw0000pes\_short.wav and m1nw0000pes\_short.wav, 16 bit, 20 kHz sampling) are also provided in the *sample* folder*.* These examples can also be used to assist in verifying whether you have a proper YAAPT setup.

## Computing pitch track with voiced/unvoiced decision for the minimum big error and then applying corrections

1. In the MATLAB command window, go the folder where the YAAPT program is located.
2. Read speech data from the sample **sample/** **f1nw0000pes\_short.wav** file.

|  |
| --- |
| >> [Data, Fs] = wavread ('sample/f1nw0000pes\_short.wav');  |

Plot the data as shown in the figure below to verify the data has been read correctly.

|  |
| --- |
| >> plot(Data); |



1. Compute the pitch track with the ***yaapt( )*** function. The computed pitch tracking is saved in an array ***Pitch*** of length ***nf****.*

|  |
| --- |
| >> [Pitch, nf] = yaapt(Data, Fs, 1, [], 0, 2);  |

 The plot of the pitch track is shown in the figure below.

|  |
| --- |
| >> plot(Pitch, '.-'); |



1. Apply corrections with ***ptch\_fix()*** function.

|  |
| --- |
| >> Pitch\_fixed = ptch\_fix(Pitch);>> plot(Pitch\_fixed, '.-'); >> ylim([0, 250]); % Set the Y axis to [0, 300] |

****

## Adjusting algorithm by passing different parameters

1. In the MATLAB command window, go the folder where the YAAPT program is located.
2. Read speech data from the sample **sample/** **f1nw0000pes\_short.wav** file.

|  |
| --- |
| >> [Data, Fs] = wavread ('sample/f1nw0000pes\_short.wav');  |

1. Use a *struct* type ***ExtraPrm*** to define the parameters that need to be modified. The optimized parameters for this case are listed in Table 1 in the work of S. A. Zahorian and H. Hu [1].

|  |
| --- |
| >> ExtrPrm.f0\_min = 60; % Change minimum F0 searched to 60Hz>> ExtrPrm.fft\_length = 4096; % Change FFT length to 8192>> ExtrPrm.dp\_w1 = 0.5; % Change DP V-V transition weight to 0.5 |

1. Compute pitch track with the ***yaapt( )*** function with modified parameters. The computed pitch tracking is saved in an array ***Pitch*** of length ***nf****.*

|  |
| --- |
| >> [Pitch, nf] = yaapt(Data, Fs, 1, ExtrPrm, 0, 2);  |

 The plot of the pitch track is shown in the figure below.

|  |
| --- |
| >> plot(Pitch, ‘.-‘); |



## Enabling plots during pitch tracking

1. In the MATLAB command window, go the folder where the YAAPT program is located.
2. Read speech data from the sample **sample/** **f1nw0000pes\_short.wav** file.

|  |
| --- |
| >> [Data, Fs] = wavread ('sample/f1nw0000pes\_short.wav');  |

1. Enable plots during pitch tracking by setting the fifth parameter (fig) to 1.

|  |
| --- |
| >> [Pitch, nf] = yaapt(Data, Fs, 1, [], 1, 2);  |

 The YAAPT creates figures to show the original speech, nonlinear processed speech, spectral pitch track, pitch candidates and final pitch track as shown below.









#  REFERENCES

1. Stephen A. Zahorian, and Hongbing Hu, "A spectral/temporal method for robust fundamental frequency tracking," J. Acosut. Soc. Am. 123(6), June 2008.
2. Stephen A. Zahorian, Princy Dikshit, and Hongbing Hu, "A Spectral-Temporal Method for Pitch Tracking," International Conference on Spoken Language Processing, Pittsburgh, PA, Sep. 2006
3. <http://www.ws.binghamton.edu/zahorian>
4. <http://pods.binghamton.edu/~hhu1/>