Contour clustering

Manual

Constantijn Kaland
University of Cologne - Institute of Linguistics

This manual provides usage instructions to run the Praat-script and R-script in order to take time-series f0 measurements and analyze them using cluster analysis, as introduced in the accompanying article (Kaland, 2021). The scripts, a tutorial, and other supplementary materials can be downloaded here:

https://constantijnkaland.github.io/contourclustering/

June 2023

Contents

1. Requirements

First, the three minimal requirements are discussed, which need to be fulfilled before the clustering approach can be applied. Although the required scripts are commented in a detailed way and provide references to the corresponding sections in this manual, a basic understanding of scripting in both Praat (Boersma and Weenink, 2022) and R (R Core Team, 2020; R Studio Team, 2020) is helpful.

1.1 Unit of analysis

The researcher is required to decide which linguistic unit of analysis needs to be investigated. The unit of analysis depends on which level of prosody is subject to the interest of the researcher (e.g. word or phrase level). For example, the procedure can be applied to investigate tonal contrasts at the lexical level as well as contours at the phrase level. In the former, the unit of analysis would typically be the syllable or word, whereas in the latter the unit of analysis would typically be the phrase. Note that some understanding of the prosodic relevance of these units in the language under investigation needs to be present. It is beyond the scope of this manual to provide fieldwork paradigms that help to identify the units that are relevant in the prosody of certain language. Studies have shown that cross-linguistic validation of prosodic units is crucial. With respect to intonation units, instructions to indicate "distinct units perceivable by means of a coherent melody" (Himmelmann et al., 2018, p. 6) lead to considerable agreement among native and non-native annotators in different languages. This would hint at the universal nature of intonation unit perception, possibly cued by boundary tones.

Selecting the unit of analysis offers an opportunity to restrict the collected data according to the demands of the research question. For example, one could select only phrase final syllables/words, only question phrases, or only phrases occurring between two silent intervals (i.e. pauses) of a specific length. In principle there is no restriction as to which unit can be analysed and the chosen unit can be further

restricted by specifying a time-range (e.g. syllables/words/phrases with a certain duration). However, the decision for a unit has a couple of implications for later steps in the procedure. For this reason, it is advisable to keep the unit of analysis the same throughout one analysis, i.e. not comparing syllable-level f0 movements with those found in phrases.

1.2 Dataset

The researcher should have access to a collection of audio-recordings and corresponding annotations in Praat textgrids (Section 1.3 for annotation requirements), consisting of sufficient observations. What a sufficient amount of observations is, will depend on the quality of the recordings and a number of later steps in the analysis. For example, when only four clusters need to be identified, less observations are needed than when eight clusters need to be identified (see Section 2.2.4 for instructions on the number of clusters). It is likely that some data needs to be discarded due to f0 measurement errors. It is therefore recommended to collect more units than strictly speaking needed. In the phrase contour example (Article section 3), 321 contours were used for initial analysis, on the basis of which at least one clearly defined follow-up hypothesis could be formulated concerning potential functional differences between in the contours. For the tonal analysis (Article section 4) the contours of 213 syllables were analysed and again provided insightful results for further testing. Another indication of whether the dataset is large enough for cluster analysis can be obtained from the standard errors within each cluster (provided in the output, section 2.2.5). Large standard errors indicate large variance within the cluster, which could be the result of (too) few observations. It is important to note that in general more observations will often lead to more representative results and that the ideal size of the dataset largely depends on decisions taken during the analysis (see Section 2.1 and 2.2). Unbalanced datasets are not necessarily problematic for the cluster analysis. For example, a dataset with 5% questions and 95% statements could be accurately distinguished assuming that their difference is marked by means of distinct f0 contours.

1.3 Segmentation

Once the unit of analysis and the dataset are decided upon, the researcher needs to segment all units of analysis on an interval tier in a Praat textgrid (Boersma and Weenink, 2022). This means that the left and right boundary of each unit

(an interval) need to be indicated. It is highly recommended to label the units using glosses in the language under investigation or another common language (e.g. English), or ideally both. The more information is available in the textgrid for each unit, the more solid the interpretation of the results can be after analysis. This information can include other linguistic aspects that are not necessary included in the initial intonation analysis (e.g. segments, syllables, words, phrases, discourse units, grammatical annotations). For a typical dataset, the researcher has access to a number of audio recordings and corresponding textgrids, which are ideally, but not necessarily, equal to the number of speakers or sessions sampled in the field or lab.

2. Workflow

In what follows, two essential components of f0 contour clustering are discussed in the form of hands-on guidelines that follow the steps a researcher is suggested to take in the initial stages of the research. These are 1) the collection of time-series f0 measurements, and 2) the statistical analysis of the f0 measurements and for generating a cluster-annotated Praat textgrid. Both components are handled by the gui version of the R-script. The first component can alternatively be run using the Praat script (potentially faster for larger datasets due to processing speed). F0 tracking is based on Modified Harmonic Sieve (Scheffers, 1983) in the R-script (Winkelmann et al., 2023) and on autocorrelation (Boersma, 1993) in the Praat-script. The R-script provides more intuitive control over the f0 smoothing accuracy using visual feedback. Note that the use of Praat remains recommended, not just for annotation using TextGrid, also for post-hoc visual inspection in the Praat editor window and for merging the generated textgrids with the original ones.

2.1 R-script: obtaining f0 measurements

Time-series f0 measurements require a dataset as described in Section 1. Measures are taken for each unit of analysis based on a combination of sound files and textgrids. Note that filenames need to be identical for the script to be able to link the textgrid annotation to the correct wavefile; only the extensions should differ (e.g. 'Recording_1.wav' and 'Recording_1.TextGrid'). The TextGrid files should not be ones that were saved as 'short text file'. Prior to running the measuring, the researcher is required to set a number of parameters that determine the way the measurements are taken. Apart from the directories that need to be specified for the respective audio files, corresponding textgrids and output-files, the parameters concerning the f0 analysis are explained in detail below.

2.1.1 Tier

The researcher needs to provide the name of the tier in the textgrid that contains the units of analysis. The measurements will be taken from the tier specified here. When multiple files and corresponding textgrids are used, the tier names containing the units of analysis should be identical across the textgrids.

2.1.2 Pitch minimum and maximum

The upper and lower boundaries of f0 calculation in Praat can be specified by the researcher. The default minimum of 75 Hz and default maximum of 500 Hz may be able to capture most f0 movements produced by the human voice. However, additional control is often desired to account for the gender of the speaker(s) or the type of phonation (Gordon and Ladefoged, 2001).

2.1.3 Time-step

The time-step setting refers to the frame duration used to calculate f0 (`wrassp::mhsF0` 'windowShift = '). The standard setting used in Praat depends on the pitch minimum (or pitch floor; see Praat Manual "Sound: To Pitch..."; Boersma and Weenink, 2022) and defaults to 10 ms with a minimum of 75 Hz. However, manually specified time-step provides the researcher another way to configure the resolution of the f0 measures. This might be needed in addition to number of measures and smoothing/stylization resolution. Note that the time-step setting determines how many measurement points are initially taken for a contour, which at the stage of f0 tracking is different from the number of measures setting (Section 2.1.5). For example, for a 200 ms interval one obtains 20 f0 measures with a 10 ms window and one obtains 40 f0 measures with a 5 ms window. This number of measurement points will be converted to number of points the user sets to estimate each contour (Section 2.1.5), i.e. after inter-, extra-polation and smoothing. If the final number of measurement poins is more than the number of point obtained using the time-step setting, the likelihood of inaccurate f0 approximation increases. That is, a contour obtained from 20 measures (e.g. 200 ms interval, 10 ms time-step) will not become more accurate with a set number of measurement points of 40. The other way around is recommended, i.e. more f0 measures than the set number of points. Thus, the time-step setting offers a way to safeguard that the tracking resolution equals or exceeds the contour resolution.

2.1.4 F0 fit

F0 is tracked using the Modified Harmonic Sieve method (Scheffers, 1983). This method aims at detecting f0 in noisy environments by a maximum likelihood estimation from its harmonics (Goldstein, 1973). The f0 fit threshold specifies the minimum probability for an f0 measurement to fit. Increasing this value makes the detection algorithm more strict, rejecting more f0 candidates and accepting only the better fitting ones (more accuracy, less candidates). Decreasing this value will accept more candidates at the expense of tracking accuracy.

2.1.5 Number of measures

The researcher needs to decide how many f0 measurements are needed to represent the smoothed contour. It is important to realise that the smoothed contour is already the product of interpolation, extrapolation and smoothing, potentially based on just a few tracked f0 points. Representing the smoothed contour by more points than actually tracked, could be misleading as this potentially generates the illusion of accurate representation (e.g. representing a smoothed f0 contour by 20 points when it is based on 5 tracked f0 points, see Section 2.1.3). It is more important whether eventual turning points in the contour are well represented by both the smoothing and the number of measurement points. This depends largely on the unit of analysis and the desired accuracy of the measures. Taking too few measures could lead to missing essential f0 movements within the contour, whereas too many measures could give too much importance to insignificantly small f0 measures, although the latter should be largely taken care of by the smoothing process. Note that the smoothing process (Section 2.1.6) or stylization process (Section 2.2.6) could also be responsible for over-estimation of small f0 movements. It is recommended to change the number of measure only in accordance with the smoothing and stylisation resolution, and the output of the cluster analysis (Section 2.3.7). It is important to note that inaccurate f0 measurements are likely to occur for each dataset. The cluster analysis provides an automatic detection of erroneous and outlying contours, which can then be removed from the data (Section 2.3.8). Changing the number of measures to account for a small number of erroneous contours is not recommended, as this affects also the correct measures. Only if the same type of f0 errors occur repeatedly, changing the number of measures could be considered.

2.1.6 Smoothing bandwith

By default, the f0 measures are interpolated and extrapolated, which means that all missing f0 points are filled in by linear interpolation and missing f0 points at the edges are filled in by linear extrapolation using a constant based on the first/last available f0 point. Thereafter, the f0 contour is smoothed using kernel density estimation (KDE). The applied method is described in Silverman (1986). The smoothing bandwith positively correlates with the degree of smoothing (i.e. larger bandwith = more smoothing). the bandwidth value is the input of the `stats::ksmooth()` function. It is recommended to try out multiple settings and inspect the results using the sampling method (Section 2.1.7).

2.1.7 Sampling

Before taking f0 measures of an entire dataset, there is an option to sample a specified number of randomly chosen intervals to inspect the accuracy of the acoustic settings. In particular the f0 floor/ceiling, number of measurement points and smoothing bandwidth should be changed to improve the quality of the output. After each sampling plots are generated showing the originally tracked f0 points (black solid line) and the interpolated, extrapolated and smoothed contour (red dashed line). Sampling provides therefore an essential quality control before doing (cluster) analysis on all the measures. It is possible that f0 cannot be tracked (accurately) in all intervals; this will be shown in the sampling plots and NAs will written to the output file.

2.1.8 Measuring

After sampling and setting the acoustic parameters to optimal values, the entire dataset can be analysed. This process generates a 'data_long.csv' file in the directory from which the R-script is run. Thereafter, the contour clustering interface can be started using this dataset and additional cleanup can be done before cluster analysis. Note that additional control over octave jumps (Section 2.2.7) can be obtained using the Praat-script.

2.2 Praat-script: obtaining f0 measurements

Note: the Praat script is no longer needed to perform time-series f0 measures. The function is built into the R-script (gui).

The Praat script works highly similar to the R-script, with additional control over octave jump correction and interval length selection. The stylization resolution setting in Praat is essentially an alternative to the smoothing process described above.

2.2.1 Tier

See Section 2.1.1.

2.2.2 Duration

The researcher has the option of restricting the analysis to units that have a length within a specified range (in seconds). This range needs to be specified prior to analysis. Setting the duration range is a means of making the set of analysed units more homogeneous. For example, if phrase-level contours are the target of analysis, the researcher can decide to measure only the most common phrase lengths (Article section 3 for an example). Note that the number of measures (Section 2.1.3) is set once and is therefore identical for all contours, thus automatically time-warping the measured contours. This provides the opportunity to compare compressed and uncompressed versions of the same contour (see Ladd, 2008, p. 180 for a discussion). Note that truncated contours are not accounted for by the duration settings. Setting the duration range may therefore be mostly needed when additional control over the unit length is desired, e.g. to exclude outlying lengths. Note that the largest degree of control in selecting a homogeneous dataset is obtained by selecting the unit of analysis (section 1.1). The script requires the duration range to be set in all cases. When no duration restrictions are needed, the range still needs to be defined and should be set using a very small number (zero is not accepted) as the minimum and a very high number as the maximum. For example, with a duration range between 0.00001 and 1000 seconds, most - if not all - intervals are taken into account for f0 measuring.

2.2.3 Number of measures

See Section 2.1.5.

2.2.4 Time-step

See Section 2.1.3.

2.2.5 Pitch minimum and maximum

See Section 2.1.2.

2.2.6 Stylization resolution

The stylisation process is handled in Praat and requires a resolution: i.e. a number indicating the minimum f0 difference between two adjacent f0 points to be taken into account in the stylised contour. The stylisation has two objectives. First, stylisation abstracts over f0 differences with a size below the specified resolution and is therefore able to discard insignificant f0 movements. Second, cluster analysis is best applied to non-missing data. In natural speech, f0 contours are frequently interrupted due to voiceless segments. It is known however, that listeners interpolate (fill in the voiceless gaps when perceiving the f0 contour) to some extent (Mixdorff and Niebuhr, 2013). Thus, by means of stylisation the researcher obtains f0 measurements that span the entire unit of analysis. The stylisation resolution is specified in semitones and a standard setting of 2 ST is recommended for initial analysis. Lower resolution values allow for more local variation in the f0 contours and could be needed to reveal phenomena that require more subtle f0 movements.

Note that with smaller numbers of measures higher degrees of stylisation are already obtained in the measurements. That is, taking 5 measures per unit of analysis could easily abstract over f0 movements that would be captured when taking 20 measures. However, with many measures, the risk increases that insignificant f0 perturbations fall within the stylisation resolution and become part of essential turning points in the stylised contour. It is highly recommended to test several combinations of number of measures and stylisation resolutions, to obtain the most accurate stylisations of the original contours. This can be done by running several rounds of measuring and cluster analyses (Section 2.3), and evaluating the accuracy of the output each time before doing any interpretation of the results. The accuracy can best be evaluatated by comparing the clustering output to the individual contours in the Praat editor window. When analysing spontaneous field data, it is likely that f0 measurement errors cannot be entirely avoided, even if the number of measures and the stylisation resolution have been set to optimal values. Therefore, erroneous cases are likely to be revealed by the cluster analysis and option to discard them are provided in Section 2.3.6.

2.2.7 Kill octave jumps

This setting allows the researcher to correct octave jumps in the units of analysis. When selected, octave jumps are removed before the stylisation process. These jumps can be the result of inaccurate f0 calculation in Praat and can be a challenge to fully correct for. This problem particularly applies to spontaneous speech and recordings made in acoustically challenging environments. When octave jumps are not handled correctly, extreme f0 movements end up in the stylised contour and will make the cluster analysis much less accurate. For example, a single contour with octave jumps might be different to all other contours in the data to such an extent that the contour forms a cluster on its own. Such an error could have an undesired effect on the outcome or at least hinders the interpretation of the results.

Although killing octave jumps is therefore recommended by default, some contours might be corrected for jumps even when there are none in the data (e.g. for instances where speakers use a wide pitch range). Furthermore, it is important to note that octave jumps are handled by assuming a correct "baseline" f0. Any jumps from this baseline will thus be lowered or heightened to this baseline. Deviations from a wrongly assumed baseline could therefore be treated erroneously as octave jumps. This could result in Praat adding or subtracting an octave to/from a large portion of the f0 contour, which would have otherwise been measured correctly. In order to obtain some insight into the octave jump handling, the pitch analysis calculates a change ratio by dividing the mean f0 of the contour after handling the octave jumps from the mean f0 of the contour before handling the octave jumps. In this way the researcher obtains a rough estimate of the overall effect of the octave jump handling on the f0 contour (written in the output file under "Jumpkilleffect"). If the change ratio decreases or increases the mean f0 to a large extent, it is recommended to inspect these cases for errors in measuring and/or octave jump handling. The option to specify the size of the allowed change is given when preparing the data for cluster analysis (Section 2.2.2). Obtaining a high number of extreme change ratios can also be a sign of inaccurate (advanced) pitch settings in Praat or problems with f0 calculation due to recording quality.

2.3 R-script: cluster (re-)analysis

The optimal settings used for clustering depend largely on the unit of analysis and the nature of the dataset. Some essential recommendations are given in this manual, however data-specific demands require careful consideration of each (combination of) setting(s). The main guidelines provided here help to recognize common problems

as f0 related (Section 2.1) or cluster analysis related (this section). However, it is crucial to note that settings used in either part of the analysis should correspond as they depend on each other for a useful outcome.

2.3.1 Optional user interface

The R-script is provided in two versions; one without graphical user interface (no-gui) and one with graphical user interface (gui) . The core elements are the same in either version of the R-script. In the no-gui version, the analysis is run directly from specific (combinations of) codeblocks in the script. Most of the settings required for usage of the no-gui version can be set in the code block "set variables" occurring at the beginning of that script. In this way, settings can be changed centrally and are intuitively visualized in the environment provided by the recommended software R-Studio (R Core Team, 2019). Note that the subsetting variables are not set centrally, but rather in their respective code blocks (section 2.2.6). The no-gui script provides additional comments explaining its code-block structure and reference to the relevant sections in this manual.

The gui version is provided using the Shiny package (Chang et al., 2020) and offers a possibility to execute the analysis without need to refer to the code. This is recommended for users who do not intend to alter the functionality of the script or do not master R scripting. The gui version also stores the most recently loaded datafile, dendrogram, table and plot automatically in the working directory until the script is stopped. The datafile is stored in two versions. A long version ('data_long.csv') essentially resembles the output of the Praat script which was uploaded in the gui (and optionally cleaned or subset). In the long version, one row represents a single f0 value. A wide version ('data_wide.csv') is stored with the most recent cluster analysis results written in an additional column ('cluster') in the dataframe. This provides a readable outcome of the cluster analysis for checking individual contours. In the wide version, one row represents a single contour, i.e. consisting of all f0 measurements that make up this contour (equal to the number of measures, Section 2.1.3). The gui version provides an option to save the current analysis (data_long, data_wide, dendrogram, table, plot, and an evaluation table and plot) for later use (see also Section 2.2.5 and 2.2.6). If the files are saved by the user, they will not be cleaned upon closing the gui version, but stored in a subdirectory called "saved" in the directory from which the script is run. The following provides guidance to using the scipts. This guidance includes explanation of the different settings the user of the gui version can choose among. Note that most of them have a setting option that is selected by default. The default settings can often be used without loosing

the core functionality. There are, however, many reasons to apply specific (other) settings for certain research questions and/or datasets, for which the explanations in the following provide guidance.

2.3.2 Selecting data for cluster analysis

The f0 movements obtained from following the procedure outlined in Section 2.1 need to be read in R. By default, the no-gui script assumes that the output file with the f0 movements is stored in the same directory as that script. Reading the dataset should be done using the correct arguments for the command 'read.csv()' in R, depending on the formatting of datafile. All the available formatting arguments needed to the read the data can be specified in the in the no-gui script. In the gui version, four arguments can be set that could overcome the most common issues; separator (default: comma, corresponding to the default output of the Praat-script), stringsAsFactors (default: false), fileEncoding (default: utf-8, change to utf-16 for more complex orthographic notations such as IPA), and SkipNul (default: false, set to true to omit nul bytes in the datafile). Note that when using the output file as directly generated by the Praat-script, the default values are likely to be the correct ones.

Prior to running the cluster analysis, it is important to obtain a dataset without missing values or other (f0) errors. Several common erroneous cases can be removed automatically from the data. The status tab in the gui version provides an overview of automatically generated statistics on the datafile. These include the number of contours in the data, the number of measurements taken (should correspond to the setting used in the Praat-script), number of speakers (equals number of different wave files), number of rows with empty filenames or labels, number of unused levels, number of f0 measurement errors. Whenever these statistics are inaccurate or reveal errors, the option 'clean data' should be selected and applied to the data. In most cases, this resolves the errors, as can then be checked from the updated statistics in the status tab. If errors remain, the datafile should be corrected manually or re-generated from the Praat-script.

The cleaning process also automatically detects spurious changes in f0. Research has shown that the rate of change in f0 has an upper limit, which is different for rises and falls (e.g. Xu and Sun, 2002, Table X). These limits are implemented in the app such that all contours within which the maximum rising rate (72 ST/s) or falling rate (96 ST/s) is exceeded, are discarded from the data.

In addition an option is provided to correct cases for which the octave jump handling resulted in deviations from the mean f0 (Section 2.1.7) that fell beyond the

range of accepted values (default: 0.90-1.10, corresponding to a maximum allowed decrease or increase of 10%). It is likely that additional data cleanup is needed before the cluster-analysis can be interpreted usefully. Note that octave jump correction is useful for measures taken in Praat and does not apply to already smoothed contours coming from the time-series f0 measures taken in the gui version of the app. Any remaining cleanup of data might be needed in terms of subsetting the data. How this remaining cleanup could be handled by the clustering technique is described in Section 2.2.6.

Furthermore, there is an option to convert the f0 values as obtained in Hertz (the default from the gui measurement interface or from Praat) to semitone or ERB is provided. The semitone and ERB scale are logarithmic which approximates the perception of pitch more accurately than the Hertz scale (see their respective differences in e.g. Nolan, 2003 and Glasberg and Moore, 1990).

2.3.3 Controlling for speaker differences

Six options are provided for handling speaker differences: 1) no correction, 2) subtracting the speaker's mean f0 from each measure, 3) standardizing the f0 measures, 4) min-max normalizing the f0 measures, 5) robust re-scaling the f0 measures, 6) octave-median re-scaling the f0 measures, or taking the first derivative (d1) of the f0 measures representing a contour. The correction methods are listed in Table 2.1.

Subtracting the mean f0 controls to some extent for speaker differences such as gender and speaking style. After applying this correction method, the dataset contains corrected f0 values centered around 0. Therefore, the corrected values could be negative. Negative values indicate that the speaker produced an f0 which is below the mean f0 of that speaker. Note that subtracting the mean f0 does not take into account speaker differences that concern f0 range. For example, a monotonous speaker will remain to show a flatter contour after subtracting the mean than a less monotonous speaker. These unaccounted differences could affect the cluster analysis. For example, consider a final rise that might be used to signal a question. This rise is likely to be shallow for monotonous speakers and steep for speakers with a more variable f0. As a result, the two types of speakers are likely to end up in to different clusters. In that case, the cluster analysis will fail to correctly group a functional difference (question signaling) due to speaker differences. Or worse, all the contours of the monotonous speaker end up in a single cluster, because the larger f0 differences between contours of speakers with a more variable f0 will be clustered first (due to complete linkage, Article section 1.6). To reduce the influence of speaker differences on the cluster analysis due to f0 range, three additional methods are provided.

Standardizing the f0 subtracts the mean f0 a speaker from each f0 measure of that speaker and divides the outcome by the f0 standard deviation of that speaker. After standardizing, the mean f0 centers around zero and the standard deviation is one. The minimum and maximum f0 values still reflect the original distribution and outlying f0 values are therefore not accounted for. Note that if outliers are the result of f0 measurement errors or stylization inaccuracies, they can still effectively be handled by means of subsetting (section 2.2.6). Standardizing is a recommended method to preserve functional f0 differences such as tone contrasts when comparing different speakers (e.g. Rose, 1987).

Min-max normalization forces all f0 measures in a scale ranging from zero to one. This is done by subtracting the minimum f0 of a speaker from each f0 measure of that speaker and divides the outcome by the f0 range of that speaker (maximum-minimum). Thus, both monotonous speakers and speakers with a more variable f0 will show comparable f0 ranges after rescaling. However, the original minimum and maximum f0 values still affect the scaling process, which might give to much weight to outlying f0 values.

To reduce undesired effects of the outliers, robust scaling could be applied. In this method the median f0 of a speaker is subtracted from each f0 measure of that speaker and divided by the interquartile range of that speaker's f0 distribution (75th quantile – 25th quantile). In this way, outliers remain outlying and can still be accounted for in the subsetting procedures. The robust scaling method does not maintain a range of values between zero and one and might therefore offer a less strict comparison between speakers with varying f0 ranges.

Another method to scale the f0 range is based on the octave scale, taking into account the speaker's median f0. This makes the f0 measures more robust to outliers and is proposed as more representative method for speech melody compared to reference values based on the mean and expressed in semitones (De Looze and Hirst, 2014) . Octave-median scaling appeared particularly accurate for estimating acoustic emphasis as a result of focus, topic change or turn-taking.

It is furthermore possible to abstract over speaker differences in f0 range and register by taking the first derivative (d1) of the time-series f0 measures that represent a contour. The first derivative essentially preserves the information on the steepness, direction and turning points (zero-crossings in d1) of the f0 contour. The first derivative has been applied to f0 in previous research as an approximation of the 'velocity profile' of a contour, which was hypothesized to be more effective for the acquisition of tonal categories than the raw f0 contour (e.g. Gauthier, Shi, and Xu, 2005).

By default, both the no-gui and gui scripts assume that each audio file (Section

2.1) corresponds to a unique speaker. If this is not the case or if all audio files are obtained from the same speaker, it is recommended to either not apply any corrections or to adjust the datafile and/or script such that speaker differences can be taken into account.

Table 2.1: Overview of speaker correction methods and their formulas. f0 = a single f0 measure (of a particular speaker), f0.spk = all f0 measures of that particular speaker.

Correction method	Formula
Subtract mean	f0–mean(f0.spk)
Standardize	f0–mean(f0.spk) / sd(f0.spk)
Min-max normalize	f0–min(f0.spk) / max(f0.spk)–min(f0.spk)
Robust rescale	f0–median(f0.spk) / 75q(f0.spk)–25q(f0.spk)
Octave-median rescale	log2(f0.spk)/median(f0.spk)
First derivative (d1)	Δf0 / Δtime

2.3.4 Distance measures

The choice for a distance (or similarity) measure determines how differences between contours are expressed numerically. Distance measures applied to time-series f0 data still need to be tested thoroughly. However, some theoretical considerations are given here based on known distance measures that have mainly been applied to time-series data in other scientific disciplines (e.g. Esling and Agon, 2012). A distinction has been made between two aspects on the basis of which similarity can be expressed numerically; shape and amplitude. As for f0 contours, shape mainly concerns the direction of the f0 movement (rising, falling, level). Amplitude concerns the width of the f0 movement and thus closely relates to the excursion size (pitch range) of the movements. Both shape and amplitude are commonly subdivided into local and global measures, depending on the domain for which similarity is sought.

Time-series distance measures have been categorised into four types (Esling and Agon, 2012); shape-based, edit-based, feature-based, and structure-based. Many of the shape-based and edit-based measures require to set additional parameters before use. This adds an extra step (and potential error) to the comparison process and one cannot always know its effect on the data beforehand. Although this overview focuses mostly on parameter-free distance measures, some others that are potentially relevant for f0 contour comparison are still discussed here.

Shape based ones compare the overall shape of the time-series' raw values and can further be separated into lock-step and elastic measures (Mori, Mendiburu, and Lozano, 2016). Lock-step ones take into account the time of each measurement point and are thus useful when no further time-warping is needed (e.g. correlation based distance measures). Although used widely as a lock-step measure, euclidean distance has limitations in its application to time-series data (Esling and Agon, 2012). Elastic measures are a more useful alternative, as they compare two time-series measures f0 contours using (dynamic) time-warping. This method is able to detect similarities even when the contours are not perfectly time-aligned. For example, when two contours show a rise-fall but one of them starts earlier than the other. Given that the representation of an f0 contour by a fixed number of measurement points (Section 2.1.5) is already time-warped, a time-warping based distance measure might thus be able to account for those differences (always requires cross-checking, see below). The question is whether to what extend time-warping based distance measures are useful when the length of the unit of analysis has been already carefully chosen. When two contours have a similar shape but differ mainly in their f0 range, e.g. a shallow rise-fall versus a steep rise-fall, some distance measures are able to detect their similarity based on scaling (e.g. uniform scaling). The question is whether this is a useful method in all research contexts. That is, the research question might be such that range differences are important features of dissimilarity between contours. In that case, scaling should be avoided.

Edit-based distance measures involve the comparison of time-series based on the number of operations needed to convert one into the other (e.g. *Levenshtein* distance). The edit-operations are handled differently depending on the measure. In particular, measures differ in how much certain edits affect the dissimilarity, and how they deal with noise or outliers in the data. A commonly used technique among these measures is the search for the longest common subsequence (LCSS).

Feature-based distance measures extract sets of features from the time-series on the basis of which similarity is computed. The extraction can be based on, for example, Discrete Wavelet Transform (DWT) or Discrete Fourier Transform (DFT). A particularly important feature of intonation contours is autocorrelation; the high correlation between subsequent measurement points. This feature not only forms the basis for f0 detection measuring (e.g. Boersma and Weenink, 2022), autocorrelation functions (ACF) can also be used to compute the difference between contours.

Structure-based measures target global similarities between time-series. This might not necessary be a useful distance measure if the aim is finding prototypical contours or (local) pitch accents. However, for contours that span an entire utterance, global trends could be informative for distinguishing contours. Structure-based

measures are further categorized as model-based, i.e. using prior knowledge to calculate the likelihood that one time-series is generated by the same model as the other, or compression-based, i.e. the idea that concatenating and compressing two different time-series lead to higher compression ratios if they are more similar. Compression Dissimilarity Measures (CDM) have been applied successfully to distinguish musical scores (e.g. Takamoto et al., 2017).

The distance measures available in the gui version are Eucledian (L2 Norm), mean absolute scaled error (MASE), dynamic time warping, pearson correlation and autocorrelation (see Table 2.2). Eucledian distance (L2 Norm) is widely used to compute the distance between two vectors due to its low computational cost and frequent availability. It is however, insensitive to misalignments in time as it is a lock-step measure and does not handle outliers well (Esling and Agon, 2012). This means that clustering outputs using Euclidean distance as a distance measure become more reliable when the data is cleaned and/or converted beforehand. Dynamic time-warping provides a way to account for misalignments between two contours that are otherwise similar in shape. These misalignments could have been the results of the time-warping that is done by taking a fixed number of measurement points to represent the f0 contour. Mean absolute squared error (MASE) is a time-series metric used to calculate the accuracy of weather forecasts (Hyndman and Koehler, 2006). This metric is scale-invariant and penalizes negative and positive distances in equal ways, outperforming related metrics such as root mean squared deviation (RMSD). Although the f0 contour comparison is different from quantifying prediction accuracy, the method is sufficiently robust to be applied on f0 time-series. For each distance calculation between two contours, one contour is taken as if it were the actual weather observation and the other contour is taken as the prediction. The output MASE value can be taken as a distance measure as it shared the assumption that at zero, there is no distance between the contours (no errors between observation and prediction). MASE values greater than one indicate that a naive forecast (assuming no change) would preform better than the given forecast values. It is likelythat f0 contour distances expressed in MASE end up being higher than one, indicating that one contour could not be predicted from the other (i.e. are highly dissimilar). Dynamic time warping as a distance measure should be avoided if the length of the unit of analysis has already been controlled for. That is, there is a trade-off to the usefulness of dynamic time-warping if all intervals have (almost) the same length. In this case, time-warping might analyse misaligned portions of the f0 contours as been similar although they are not. The importance of time-warping depends thus on the research question and data at hand. Pearson correlation distance is an effective lock-step measure to compute overall similarity between contours that are represented by

time-series data. It is important that Pearson correlation coefficients are computed in such a way that they range from -1 to 1 (maximum negative and maximum positive correlation respectively). If direction of the f0 contour is not taken into account (as by taking the absolute correlation coefficient), the distance measures will regard simultaneously diverging contours as being similar. The computation as implemented in the gui version takes direction into account and is therefore suitable for application on f0 contours. Autocorrelation - being a feature-based measure - is effective to deal with f0 contours as it takes into account that adjacent f0 points in a contour are generally highly correlated. The autocorrelation distance is computationally costly.

Note that for the computation of correlation coefficients, time-series values are expected to have a standard deviation that is higher than zero. This means that flat contours will obstruct the computation of correlation coefficients. The gui version offers an option to deal with this problem by applying a declination effect to contours that are initially represented as being flat (declination correction . This is done by calculating the duration of the contours (interval length) and then apply the declination formula for speech intervals shorter than 5 seconds in Hart, Collier, and Cohen (1990) (p.129). In this way, "flat" contours will also exhibit variation in their f0 that is naturally expected. Note that the reasons for f0 to be initially represented as flat may vary, but it is likely the result of f0 tracking and/or smoothing. The number of measurement points also has an influence. These settings should be cross-checked if many contours are represented without standard deviation.

Table 2.2: Overview of available distance measures and the respective R package used for computation.

Distance measure	R-package	pros	cons
Eucledian (L2 Norm)	stats	widely used	insensitive to misalignment
mean absolute scaled error (MASE)	Metrics	outperforms RMSD	not widely used for f0
dynamic time warping	proxy	sensitive to misalignment	overcorrection
pearson correlation	TSdist	overall similarity	contour cannot be flat
autocorrelation	TSdist	high potential for f0	computationally costly

2.3.5 Linkage criteria

With hierarchical agglomerative clustering there are multiple ways in which the clusters can be formed. Note that the hierarchical clustering used in the current approach is bottom up, thus starting with each observation in a separate cluster. Clusters are formed by merging existing clusters until there are only two clusters left. Theoret-

ically, one extra step applies in that the final two clusters are merged such that all observations are in a single cluster. However, this final step is tantamount to no clustering and is not informative given that clustering is meant to obtain subgroups of observations in a dataset. Which clusters get merged is determined by the *linkage criterion* . The linkage criterion specifies how distances (dissimilarities) between clusters are computed. An overview of the linkage criteria implemented in the gui version are listed in Table 2.3. The default one recommended here is complete linkage, given the idea that maximal acoustic contrasts is the principle often taken to underlie the composition of vowel inventories (e.g. Lindblom, 1986). There is no a priori objection to select a different linkage criterion, depending on the type of f0 variation in the data.

Table 2.3: Overview of linkage criteria

Linkage criterion	Cluster dissimilarity definition
Complete	Computes all pairwise dissimilarities between the observation(s) in the clusters and takes the maximal dissimiliarity. Clusters with the smallest maximal dissimilarity get merged, leading to maximal intercluster dissimilarity.
Single	Computes all pairwise dissimilarities between the observation(s) in the clusters and takes the minimal dissimiliarity. Clusters with the smallest minimal dissimilarity get merged, leading to minimal intercluster dissimilarity.
Average (UPGMA)	Computes all pairwise dissimilarities between the observation(s) in the clusters and takes the mean dissimiliarity. Clusters with the smallest mean dissimilarity get merged.
Ward	Computes the increase in sum of squares for all possible pairwise merges of clusters. Clusters with the smallest increase get merged. This method keeps the total within-cluster variance minimal. Ward.D and Ward.D2 differ in that the cluster differences are squared in the latter and therefore emphasized, leading to easier cluster differentation.
McQuitty (WPGMA)	Similar method to average linkage, without considering the number of observations in a cluster and with taking into account the similarity between the most recently merged clusters.
Centroid (UPGMC)*	Computes the centroids (central point of all observations, i.e. a vector of means) of the clusters. Clusters with the centroids closest to each other get merged.
Median (WPGMC)*	Similar to centroid linkage, with taking into account the similarity between the most recently merged clusters.

* These linkage criteria have the risk of leading to inversion. For most link-

age criteria, merging happens on the basis of iteratively larger dissimilarities. With centroid-based linkage, centroid distances could get smaller in subsequent merging iterations. In such case, it is no longer possible to assume that with an increasing number of clusters variance among the observations in a cluster increases, whereas variance among the clusters decreases (see within and between cluster variance discussed in Section 2.3.9).

2.3.6 Cluster analysis / number of clusters

Hierarchical clustering provides the researcher with a dendrogram, a tree-structure showing the outcome of each merge. The dendrogram provides a first visualisation of the clustering process. On the basis of the dendrogram the researcher can decide on the amount of clusters. Finding the ideal number of clusters is the essential component in the approach outlined in the article. The dendrogram also provides insight in the scale of the f0 differences between the clusters. Since the largest numerical differences between clusters are found at the top of the dendrogram, an initial analysis might reveal only the differences involving a larger f0 range. In such an initial outcome, the dendrogram is likely to show asymmetry. This asymmetry is the result of late adjoining of major f0 excursions (e.g. large boundary tones) with a cluster containing all smaller scale (e.g. phrase-internal) f0 excursions. Asymmetry can be avoided to some extent by choosing a speaker correction method that accounts for outliers. Initial clustering outcomes thus provide insight into contour differences of the largest scale. Smaller scale differences, such as f0 peak height, are more likely to be successfully revealed when increasing the number of clusters. However, large numbers of clusters might result in clusters consisting of few contours, compromising the prototypicality of these contours. To accurately reveal small scale f0 differences, it is recommended to analyse a more homogeneous (controlled) dataset, or a subset of the data leaving out large-scale f0 excursions. The dendrogram provides useful directions for the degree and type of subsetting needed, as further discussed in Section 2.2.6.

Before deciding on the number of clusters, it is recommended for the researcher to have a theoretically motivated estimation of this number, i.e. before obtaining any result from the analysis. For example, if the aim is to find a basic set of different lexical tone contours from words, around four to six clusters could be sufficient to accurately capture the contrasts. However, if the aim is to find a broad set of all lexical tone contrasts in combination with phrase intonation patterns, e.g. 15 clusters could be the minimum number needed. One way of determining the number of clusters is to run several rounds of analysis, each time with an increasing

number of clusters (Article section 2, 3 and 4). This is particularly useful in an exploratory context, where only a rough estimation of the number of clusters can be made. It is also recommended to run the cluster analysis with a number of clusters that exceeds the hypothesized number. In this way, the researcher can reduce the risk of missing relevant contrast that were unexpected. The dendrogram provides the initial guidance in finding the ideal number of clusters for a certain analysis. Obtaining the dendrogram for a given dataset does not require any prior decisions on the number of clusters. That is, due to the hierarchical clustering technique used here, a dendrogram can be obtained before the number of clusters is set. Commonly, the dendrogram shows asymmetry as the result of a cluster with a single observation being merged with a cluster with many other observations. If the aim is to find prototypical f0 contours, it is questionable whether single-observation-clusters contribute to this aim (see also Section 2.2.5). They might constitute either outliers that fail to represent the dataset or unique cases which the researcher could consider for further individual analysis. It is not recommended to remove small clusters from the dataset without inspection. The subsetting procedures described in Section 2.2.6 can reduce the asymmetry in the dendrogram. An indication of the accuracy of the number of clusters can also be derived from plotting the mean contours per cluster, as further explained below. Apart from the general guidelines outlined here, there are statistical methods to obtain an estimation of the ideal number of clusters in an analysis, which are left for the user to explore (e.g. Charrad et al., 2014).

2.3.7 Interpreting the output

Once the researcher has settled on a number of clusters for initial analysis, the output of the cluster analysis can be used to create a number of essential data summaries that guide the interpretation. Apart from the dendrogram, the current approach provides three summaries; a table, a plot and an automatic annotation.

The table provides 1) the number of contours in each cluster, 2) the mean standard error per cluster, as calculated by adding up the standard errors for each measurement point and divide the outcome by the number of measurement points, and 3) a tentative indication (flagging) whether a cluster should be treated with caution. The flagging is given by a zero (no caution) or a one (caution advised) based on the data obtained in (1) and (2). That is, caution is advised when a cluster contains only one contour or when the mean standard error is more than two times the median of the mean standard errors of all clusters. The logic behind the latter threshold is to obtain a single criterion that can be applied to f0 values regardless of the speaker

correction method or number of clusters chosen (Section 2.2.3). In particular, standard errors might be affected by some of these correction methods. By taking the median as center value, the distance from zero (no deviation) is known. Thus, mean standard error values that lie further away from median than zero does, are advised to be treated with caution. The criterion is a rough estimate of how deviant the mean contour in a particular cluster is. Individual inspection is still advised to be able to quantify the standard errors on a known f0 scale (e.g. Hertz or semitones). Therefore, the inspection should be done before applying speaker correction. For example, a single cluster with a mean standard error of 10 Hz might already be enough to overlook potentially meaningful differences between the contours. That is, the f0 range for human speech can be taken as 75 to 500 Hz (standard setting in Praat; Boersma and Weenink, 2022). Thus, allowing for an average deviation of up to 10 Hz for the entire contour corresponds to a semitone difference that lies between 0.3 (480-500 Hz) and 2.2 (75-85 Hz). It has been shown that in Dutch, for example, 1.5 ST excursions (locally!) can be enough to perceive a linguistically meaningful prominence shift (Rietveld and Gussenhoven, 1985). Although the generalizability of this finding to other languages is open to further research, it is crucial to note that within cluster variance of 10 Hz might not fully avoid overlooking important f0 movements in the contour. The double-median threshold for the mean standard errors could therefore be too crude without consideration of the f0 scale. While smaller standard errors are generally preferred, increasing the number of assumed clusters can help to gain insight into this type of variability and is recommended prior to any subsetting procedures. The standard errors can thus be taken as an indication of how well the contours fit in the cluster. As a rule of thumb: if homogeneously sized clusters with a low mean standard error can be achieved with subsetting applied only to discard erroneous cases, the analysis is likely to provide an optimal outcome.

The plot provides a line through the mean f0 values for each measurement in each cluster and draws a shaded area around the line indicating the standard deviation. In addition, the number of observations in each cluster, as also given in the distribution table, are plotted for reference. The plot can then be used to link a prototypical contour onto an individual observation. It is recommended to check individual observations in order to inspect the small clusters, and to obtain immediate feedback on the accuracy of the f0 measures and the chosen number of clusters. If only minimal differences between the clusters are found, the f0 contours might have been too much stylised and/or the number of clusters too high. If the plotted mean contours show a large number of local f0 changes (i.e. not smooth, large standard deviations), the f0 measurements could have been too fine-grained, possibly magnifying the differences between the clusters and failing to accurately approach the prototypical shape of a

contour (see Section 2.1 for a discussion on the relevant settings).

It is plausible that some of the clusters in the plot show highly similar contours. This is to be expected with a (recommended) high number of clusters for initial analysis. It is recommended to not solely rely on the plot of mean contours to inspect the potential similarity between two clusters. It might not be immediately clear why two similar contours end up in separate clusters. Inspection of the actual acoustic differences between the contours is recommended in such a case. This can be done by reading the f0 values from the plot. These values should be compared to more obvious differences between other clusters obtained from the same analysis. For example, rising phrase final boundary tones could have a large f0 range and therefore be visually easy to detect. This type of "landmark" in the contour can provide an indication of the scale of the differences between clusters. Thus, visually similar contours in separate clusters might also be the result of small numerical differences which could be more accurately clustered after subsetting. Decreasing the number of assumed clusters could also be considered as a means to obtain visually more distinguished contours (an example of such a consideration is given in Article section 3). However, this method makes the cluster analysis more course-grained, introducing a higher risk of overlooking relevant contour differences.

For more thorough interpretation of the clustering analysis, two functions are provided: saving the dendrogram, data files, table and plot (gui version) and generating textgrids. Saving the files as the output of a particular cluster analysis is useful for future reference and further inspection and avoids redoing the cluster analysis once the number(s) of clusters are settled on. Generating textgrids essentially indicates for each contour to which cluster it belongs. The generated output is a textgrid file with one tier that can be merged with the existing textgrid that contains the interval tier from which the unit of analysis was taken for f0 measures (Section 2.1). With the plotted mean contours as a reference, the researcher can now go through the wave file and textgrid in Praat, and explore the potential meanings of each prototypical intonation contour. It is crucial to note that the cluster labelling is likely to differ when the cluster analysis is re-run with different settings (i.e. number of clusters, cluster technique, different ways of analysing f0 etc.). It is recommended for the researcher to try different settings and repeatedly go back and forth between running the analysis and interpreting the outcome. There is no a priori indication of how many times the analysis needs to be run for the researcher to obtain an informed interpretation of the data, and to ultimately be able to attribute specific functions to specific contour shapes. A typical analysis could consist of an initial run on all data and subsequent runs on informed subsets, as a means of "zooming" into cluster differences that would otherwise remain masked due to asymmetry (see the article;

Section 1.6 for a theoretical motivation and Section 2 and Section 3 for examples). However, much depends on the type of speech data or contours under analysis, and the language at hand.

The automatic annotation allows the researcher to compare the original contour in Praat with the mean contour representing a cluster that was generated by the analysis. Interpreting the automatic annotations could therefore reveal f0 measurement errors. It is plausible that a contour is assigned to a cluster without actually sharing similarities with the plotted mean contour of that cluster. In this case, f0 measurement errors are the most likely cause of the mismatch. Although these errors cannot be avoided in all cases, it is recommended to inspect them individually and improve f0 measurement settings or change the number of clusters.

2.3.8 Subsetting

A procedure for subsetting the data is provided in the no-gui and gui version. This procedure has two purposes. Its initial use can be applied to a high number of clusters (Article section 3 and 4 for examples) in order to remove erroneous or outlying contours. For this purpose, a high number of clusters (e.g. 25 or more) is recommended to obtain small sized clusters. If the data consists of erroneous or outlying contours, these will then be revealed in small sized clusters, potentially single-contour clusters. Thus, when removing small sized clusters, the risk of discarding error-free contours remains low. This way of "pruning" the data can be done using the automatic flagging of erroneous/outlying contours (Section 2.2.5).

A second purpose of the subsetting is to "zoom in" into a specific subset of the data, typically after initial round(s) of clustering. For example, initial clustering could reveal a small number of rising contours among an overall majority of falling contours. It could be useful to separate the rises and falls in subsets and perform subsequent clustering on either subset. This has the advantage of revealing smaller scale differences between the contours (e.g. different types of rises or falls), without their differences being affected by contours of the other category. Such an application of the subsetting procedure is particularly useful when there is clear indication or supportive evidence from distinguishing these two types as categories.

In the non-gui version, an additional automatic subsetting procedure is provided. This procedure is based on setting a minimum portion of the data to be left after subsetting. Then, the smallest cluster is removed in each round of cluster analysis until the minimum portion is reached. This subsetting procedure provides a crude means of removing non-prototypical contours, particularly useful for large datasets with a high degree of asymmetry in the dendrogram. Note that this procedure should

only be run in the Rstudio environment (R Studio Team, 2020), in order to keep track of the outcome. In particular, as the risk exists that non-erroneous contours will be removed from the data.

A general word of caution should be given here, as subsetting essentially ignores (potentially large) parts of the collected data. This procedure compromises the representativeness of the empirical investigation and re-introduces the risk of giving researchers' intuitions a decisive role in the analysis-process. Although these disadvantages cannot be entirely avoided, it is crucial to keep the ultimate goal of the investigation in mind. The procedures outlined here are designed to reveal prototypical contours, for which some deviant instances can be naturally expected in spontaneous speech (see further discussion in the Article section 3).

2.3.9 Evaluation

The gui version provides an additional interface to evaluate the ideal number of clusters. Two methods are implemented; one based on information cost (Shannon, 1948) and minimum description length (MDL, Rissanen, 1978), and one based on within and between cluster variance.

The first evaluation method computes information cost of three aspects of the cluster analysis; 1) the cost of specifying the clusters, which is expected to increase with more clusters, 2) the probability of a contour being in a specific cluster, given what the prototype (mean) of that cluster is, which is likely to decrease with more clusters, and 3) the cost of specifying each contour within a certain cluster, this is likely to increase with more clusters, depending on how well an individual contour fits the cluster (see details and demonstration in Kaland and Ellison, 2023). The summed outcome is then taken as a single evaluative information cost measure for one round of cluster analysis (one 'clustering'). When applied over a range of clusterings, e.g. starting with two clusters up to ten clusters, the evaluation measures for each clustering round are likely to produce a U shaped curve when the number of clusters are plotted on the x-axis and the information cost measure on the y-axis. The gui version provides a tab panel to perform this type of evaluation. The user is required to specify the range of cluster rounds and a bending factor (measurement point dependency value). The former indicates the number of clusters that should be assumed for the first and the last round of cluster analysis that should be evaluated. The evaluation then runs those rounds and all rounds with intermediate numbers of clusters. The latter value indicates how dependent adjacent measurement points are, with higher values corresponding to higher degrees of dependency. Note that this is a way to take into account the length of the unit of analysis, which could range

from (portions of) syllables (high dependency) to entire utterances (low dependency). The bending factor is suggested automatically, usually within a range of 1 to 5. The automatic suggestion is based on rough ranges of the mean duration of the unit of analysis (in seconds, see Table 2.4). Suggested values can be manually overridden and can be specified using values (with decimals, if needed). This might be needed to obtain a clearer U shaped bend in the plotted curve, but should be informed by the unit of analysis and the absolute time-gap between measurement points. When analysing a dataset for which time-normalization has been done outside the provided Praat- or R-scripts, the bending factor needs to be adjusted accordingly, potentially using different values than suggested here.

The evaluation procedure generates a plot with the information cost for each clustering round. The clustering round with the lowest information cost matches the MDL and could be taken as the ideal number of clusters for the dataset under analysis. Note that all evaluation is based on the specific dataset under analysis and does therefore not directly allow generalization for the contour inventory of that specific language.

Second, the method based on within and between cluster varaition quantifies the variance based on the f0 values (either converted, speaker-corrected, both or unconverted Hz values). It is expected that within cluster variance of the f0 contours decreases with more clusters, because with more clusters the individual contour within a certain cluster are more alike than with less clusters. In addition, between cluster variance is expected to increase with more clusters as the mean f0 contour in a certain cluster will differ more from the mean f0 contours in other clusters (i.e. more varying mean contours with more clusters). Note that these principle do not necessarily apply when chosing centroid-based linkage criteria (inversion), see Section 2.3.5. Theoretically, the optimum clustering is the one for which the lowest within cluster variance and the highest between cluster variance is observed. With both types of variance being inversely correlated, the optimum will thus be around the cross-over point where both types of variance are the closest to each other, assuming identical scales for both types of variance.

Within cluster variance is computed by taking the standard deviation of each measurement point iteratatively for all measurement points in a cluster. That is, starting with cluster 1, the standard deviation is taken from all the first measurement points in that cluster, then the standard deviation is taken from all the second measurement points in that cluster (etc.), until the final measurement point in that cluster. Then, the mean of all the standard deviations is taken to represent the variance in that cluster, after which the same procedure is repeated for the remaining cluster(s). Thereafter, all the mean standard deviations (one per cluster) are

averaged again to represent within cluster variance for one clustering.

Between cluster variance is computed by taking the mean f0 value of a measurement point across clusters. That is, starting with measurement point 1, their mean value in each cluster is taken iteratatively for all clusters. Then, the absolute difference between the maximum and the minimum of these mean values is taken to represent the between cluster variance of that particular measurement point. This procedure is repeated for all measurement points and the single value representing the between cluster variance of a particular clustering is the mean of all these difference measures.

In order to make the within and between cluster variance comparable, all the values are scaled between 0 and 1. The scaling is done for each evaluation round. For example, if an evaluation round is set to compare clusterings ranging from 2 to 10 clusters (9 clusterings), the resulting 9 values representing within cluster variance as well the 9 values representing the between cluster variance are scaled. In this way, the cross-over point can be plotted in a graph where both types of variance share the same y-scale (0-1).

Note that the cluster variance evaluation method is sensitive to the range of clusterings taken into account. That is, the scaled variance might vary to some extent depending on the range of clusterings. It is recommended to observe the relative gain or loss in the curves representing within/between cluster variance. If a ceiling or floor effect can be observed, this could be taken as an indication that extending the maximum of the selected range is no longer needed. Rather, if the ceiling or floor is stretched over multiple clustering rounds at the high end of the range, the scale of cluster variance might be stretched, such that the point indicating the optimal number clusters moves as a result. This is in principle undesired and does not necessarily happen for each dataset, i.e. the output might be robust enough to remain unaffected. Nevertheless, it is recommended to have an informed idea about the maximum number of clusters to test in the evaluation.

The two evaluation methods do not need to show corresponding results. It is important to note that the approaches have different backgrounds and use therefore different scales to express the optimal number of clusters. One method seeks the optimum in terms of informativity of describing the dataset. The other method provides insight into the clustering process and how the contour variation is distributed within or between clusters. That is to say that they can both provide useful perspectives on what the number of clusters to choose could be.

Table 2.4: Mean durations of unit of analysis and suggested bending factor.

M duration unit of analysis (s)	Suggested bending factor	Approx. unit of analysis
>1	5	(long) utterances
0.7-1	4	phrase(s)/utterance
0.45-0.7	3	word(s)/phrase
0.15-0.45	2	syllable(s)/word
<0.15	1	syllable (portion)

3. References

Boersma, P. (1993). Accurate Short-Term Analysis Of The Fundamental Frequency And The Harmonics-To-Noise Ratio Of A Sampled Sound. *Proceedings of the Institute of Phonetic Sciences*, *17*, 97–110.

Boersma, P., & Weenink, D. (2022). Praat: Doing Phonetics by Computer. http://www.praat.org/

Charrad, M., Ghazzali, N., Boiteau, V., & Niknafs, A. (2014). NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set. *Journal of Statistical Software*, *61*(6). https://doi.org/10.18637/jss.v061.i06

De Looze, C., & Hirst, D. (2014). The OMe (Octave-Median) scale: A natural scale for speech melody. *7th International Conference on Speech Prosody 2014*, 910–914. https://doi.org/10.21437/SpeechProsody.2014-170

Esling, P., & Agon, C. (2012). Time-series data mining. *ACM Computing Surveys*, *45*(1), 1–34. https://doi.org/10.1145/2379776.2379788

Gauthier, B., Shi, R., & Xu, Y. (2005). Recognising Tones by Tracking Movements-How Infants May Develop Tonal Categories from Adult Speech Input. *ISCA Workshop on Plasticity in Speech Perception (PSP 2005)*.

Glasberg, B. R., & Moore, B. C. (1990). Derivation of auditory filter shapes from notched-noise data. *Hearing Research*, *47*(1-2), 103–138. https://doi.org/10.1016/0378-5955(90)90170-T

Goldstein, J. L. (1973). An optimum processor theory for the central formation of the pitch of complex tones. *The Journal of the Acoustical Society of America*, *54*(6), 1496–1516. https://doi.org/10.1121/1.1914448

Gordon, M., & Ladefoged, P. (2001). Phonation types: A cross-linguistic overview. *Journal of Phonetics*, *29*(4), 383–406. https://doi.org/10.1006/jpho.2001.0147

Hart, J. '., Collier, R., & Cohen, A. (1990). *A perceptual study of intonation: An experimental-phonetic approach to speech melody* [OCLC: 708567537]. Cambridge University Press.

Himmelmann, N. P., Sandler, M., Strunk, J., & Unterladstetter, V. (2018). On the universality of intonational phrases: A cross-linguistic interrater study. *Phonology*, *35*(2), 207–245. https://doi.org/10.1017/S0952675718000039

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

Kaland, C. (2021). Contour clustering: A field-data-driven approach for documenting and analysing prototypical f0 contours. *Journal of the International Phonetic Association*, 1–30. https://doi.org/10.1017/S0025100321000049

Kaland, C., & Ellison, T. M. (2023). Evaluating cluster analysis on f0 contours: An information theoretic approach on three languages. *Proceedings of the 20th International Congress of Phonetic Sciences*.

Ladd, D. R. (2008). *Intonational phonology* (2nd ed). Cambridge University Press.

Lindblom, B. (1986). Phonetic universals in vowel systems. In J. Ohala & J. Jaeger (Eds.), *Experimental Phonology* (pp. 13–44). Academic Press.

Mixdorff, H., & Niebuhr, O. (2013). The Influence of F0 Contour Continuity on Prominence Perception.

Mori, U., Mendiburu, A., & Lozano, J., A. (2016). Distance Measures for Time Series in R: The TSdist Package. *The R Journal*, *8*(2), 451. https://doi.org/10.32614/RJ-2016-058

Nolan, F. (2003). Intonational equivalence: An experimental evaluation of pitch scales. *Proceedings of the 15th international congress of phonetic sciences*, *39*, 771–774.

R Core Team. (2020). R: The R project for statistical computing. Retrieved July 11, 2019, from https://www.r-project.org/

R Studio Team. (2020). RStudio: Integrated Development for R. Retrieved July 11, 2019, from https://www.rstudio.com/

Rietveld, A., & Gussenhoven, C. (1985). On the relation between pitch excursion size and prominence. *Journal of Phonetics*, *13*(3), 299–308. https://doi.org/10.1016/S0095-4470(19)30761-2

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*(5), 465–471. https://doi.org/10.1016/0005-1098(78)90005-5

Rose, P. (1987). Considerations in the normalisation of the fundamental frequency of linguistic tone. *Speech Communication*, *6*(4), 343–352. https://doi.org/10.1016/0167-6393(87)90009-4

Scheffers, M. T. M. (1983). Simulation of auditory analysis of pitch: An elaboration on the DWS pitch meter. *The Journal of the Acoustical Society of America*, *74*(6), 1716–1725. https://doi.org/10.1121/1.390280

Shannon, C. E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal*, *27*(3), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Chapman & Hall/CRC.

Takamoto, A., Umemura, M., Yoshida, M., & Umemura, K. (2017). Improving Compression Based Dissimilarity Measure for Music Score Analysis [arXiv:1710.01446 [cs, eess]]. Retrieved October 10, 2022, from http://arxiv.org/abs/1710.01446

Winkelmann, R., Bombien, L., Scheffers, M., & Jochim, M. (2023). *Wrassp: Interface to the 'ASSP' library* (manual). https://CRAN.R-project.org/package=wrassp

Xu, Y., & Sun, X. (2002). Maximum speed of pitch change and how it may relate to speech. *The Journal of the Acoustical Society of America*, *111*(3), 1399–1413. https://doi.org/10.1121/1.1445789

4. Index