

# Automated Spectrogram Analysis for Meteor Head Echoes

C. Powell<sup>1</sup>

The Meteor Echo Spectrogram Analysis (MESA) program is developed to automatically identify meteor head echoes in spectrograms generated by forward-scatter radio meteor detection. The program is both extensible and flexible, allowing calculations such as line-of-sight approach and recede velocities, duration, and maximum intensity. A detection sensitivity of 0.790 is achieved, with an improved sensitivity of 0.875 when only low-noise spectrograms ( $\sim \frac{2}{3}$  of the data) are analysed. The MESA program cannot fully replace manual analysis of spectrograms, but greatly reduces the volume of data that needs processing.

Received 2019 February 9

## 1 Introduction

The aim is to produce a computer program that takes spectrogram images generated by forward-scatter radio meteor detection as input, and identifies meteor head echoes in the spectrogram. The program may also make an estimate of the approach & recede velocity, maximum intensity, start time, end time, and duration, of each identified meteor echo.

The computer application uses a number of image processing techniques applied in a manner motivated by the characteristic shape of meteor echoes in spectrograms, to identify the ‘centre’ of each echo based on a centre of mass algorithm, which uses RGB pixel values as weights. Velocities may then be estimated using the Doppler equation and identification of the ‘start’ and ‘end’ of each echo, again using image processing techniques.

In order to verify the accuracy of the program, results are compared to manual calculations of a random selection of data over a timespan of 1 week to determine the sensitivity of the program.

As with all methods of visualising data, spectrograms lose some of the original data (i.e. the radio signal from detection is not kept). However, spectrograms are often the only practical way that data can be archived for an observer - no ‘level 1’ data is available, so automated spectrogram analysis is a useful tool.

## 2 Background

The theory behind forward-scatter radio meteor detection and the resulting meteor echo characteristics in spectrograms is discussed, as well as necessary theory for velocity estimation, and past work to identify meteor echoes in spectrograms.

### 2.1 Spectrogram generation

Detecting meteors using forward-scatter radio uses a transmitter broadcasting over a wide area of sky, through which meteors pass. The ionised trail of a meteor reflects the radio signal, which can then be picked up by a receiving station.

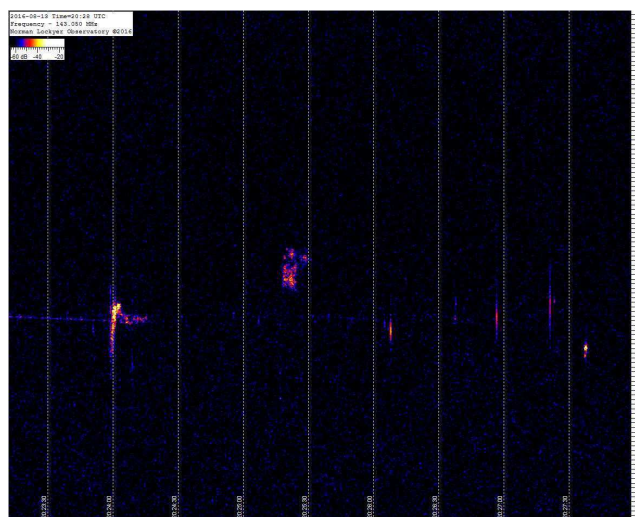


Figure 1 – Spectrogram generated by Spectrum Lab v3.0

Programs such as Spectrum Lab<sup>a</sup> receive the signal generated by radio detection and produce spectrograms (Figure 1), which are three-dimensional representations of the data: the horizontal axis is time, the vertical axis is (modulated) frequency, and the third axis is signal intensity, represented using a colour scale. Often the CMRmap colour scale is used (Figure 2). Spectrograms are sometimes referred to as ‘waterfall’ plots.



Figure 2 – CMRmap colour scale

Spectrograms most often cover a 5-minute interval, so a single detection station may generate 288 spectrograms per day: manual analysis of this is not practical, hence automated analysis is necessary.

### 2.2 Meteor echo characteristics

Assuming upper sideband forward-scatter radio detection is used, and the spectrogram uses a CMR colour scale, meteor echoes appear as white (high intensity) vertical streaks, often with highest intensity close to the carrier wave (CW) frequency. Note that often the

<sup>1</sup>Email: cpowell@cwpa.io

<sup>a</sup>Available at <http://www.qsl.net/d14yh/spectra1.html>

CW frequency is demodulated and shifted, so that the direct signal, when visible, does not appear in the spectrogram at the transmitted CW frequency.

The CW frequency corresponds to zero line-of-sight (LOS) velocity of the meteor. Greater frequencies than CW correspond to greater LOS approach velocity, and lower frequencies than CW correspond to greater LOS recede velocity. Velocity can be (naïvely) calculated using the Doppler equation

$$\frac{\Delta f}{f} = \frac{v}{c} \quad (1)$$

where  $c$  is the speed of light in a vacuum. This assumes a simple structured echo; more complex echoes require more detailed techniques for velocity estimation. Note also that the Doppler shift is in fact made up of two parts; a shift owing to the radial velocity between the meteor and transmitting station, and a shift owing to the radial velocity between the meteor and receiving station. Only the Doppler shift of the former radial velocity is considered, limiting the accuracy of the velocity estimate.

Some radar systems receive a direct signal from the transmitter, meaning the CW frequency (after shifting/demodulating) is immediately obvious. For other systems the resultant CW frequency may be determined by averaging many images together until the direct signal is apparent.

### 2.3 Past work

Echo-counting software has been developed before, for example Noguchi & Yamamoto (2008) developed an image processing tool to automatically analyse HROFFT spectrograms however no capacity for further calculations is included. A 90% agreement between manual and automatic counting was found. Artificial Neural Networks have also been used to attempt to automatically identify echoes in spectrograms (Roman & Buiu, 2015).

## 3 Design

An example spectrogram (Figure 1) is used to show each step of the processing & analysis.

### 3.1 Data assumptions

The MESA program assumes that data is provided with a size of  $800 \times 1133$  pixels, and the CMRmap colour scale is used, with a detection setup using upper-sideband forward-scatter radio detection. However, settings can be modified to remove these assumptions and allow use of any colour scale and image dimensions necessary. Namely, the number of pixels to be removed on each edge (to remove elements such as the legend and frequency scale), and the RGB channel to be extracted. The program may also be modified to use lower-sideband radio detection.

### 3.2 Data pipeline

It is envisaged that the automated echo detection system is part of a more complex data pipeline where

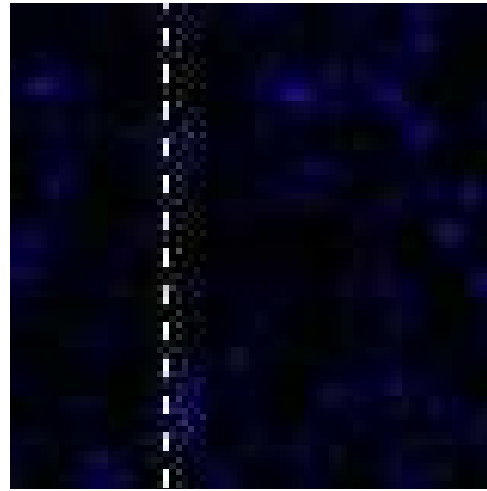


Figure 3 – JPEG compression artefacts

large, high-quality images are passed to this automated detection system where they are processed and subsequently saved to an archive in a lower quality format to reduce the necessary amount of memory storage. This removes issues raised by compression artefacts, for example in JPEG compression (Figure 3). If lossless compression is used prior to processing, this pipeline is not necessarily needed.

### 3.3 Pre-processing

Irrelevant data must be removed from the image to aid the analysis, namely the colour scale, legend, frequency scale, and timestamps. The user must enter a depth in pixels for each border of the image so that these can be removed. Spectrograms produced by Spectrum Lab also have vertical dashed bars indicating 30 second time increments, which must be removed. Since use of the CMRmap colour scale is assumed, only the red RGB channel is extracted from the image. The blue and green channels hold redundant data: most noise appears blue, whilst echoes are red to white. The image is also stretched with horizontal scale-factor 3, for reasons discussed in Section 3.5, producing Figure 4.

The colour scale must also be identified within the image to allow calculation of maximum intensity for each meteor echo.

### 3.4 Denoising

Before echoes can be identified in the image, noise must be removed. There is often a large amount of noise throughout the image, as can be seen from the blue spots in Figure 1. The denoising process uses thresholding, erosion, a total variation filter, and dilation.

The image is first dynamically thresholded using the Otsu method (Otsu, 1979). This is a common first step in image segmentation algorithms, and is a clustering-based threshold method which minimises the intra-class variance of the image's gray-level histogram, splitting the image into 'foreground' and 'background'. The background is removed by replacing every 'background' pixel with value 0. The result is an image with significantly reduced noise, Figure 5.

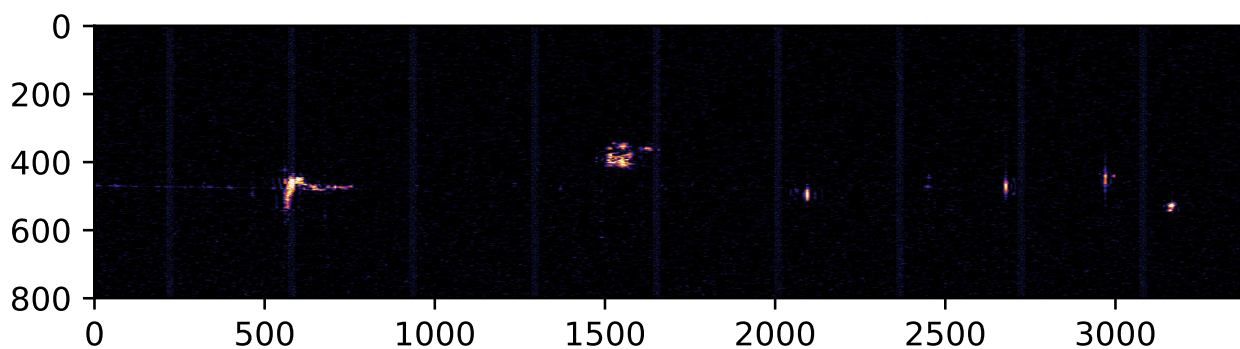


Figure 4 – Spectrogram after pre-processing.

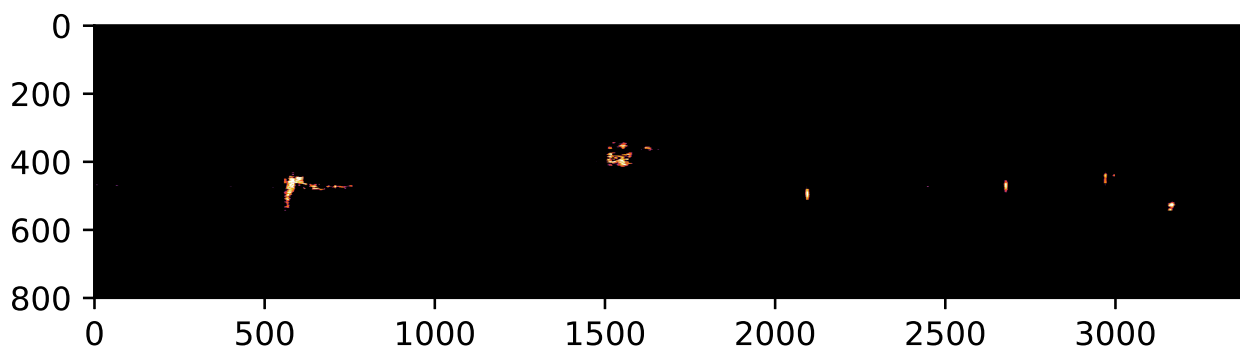


Figure 5 – Spectrogram after thresholding.

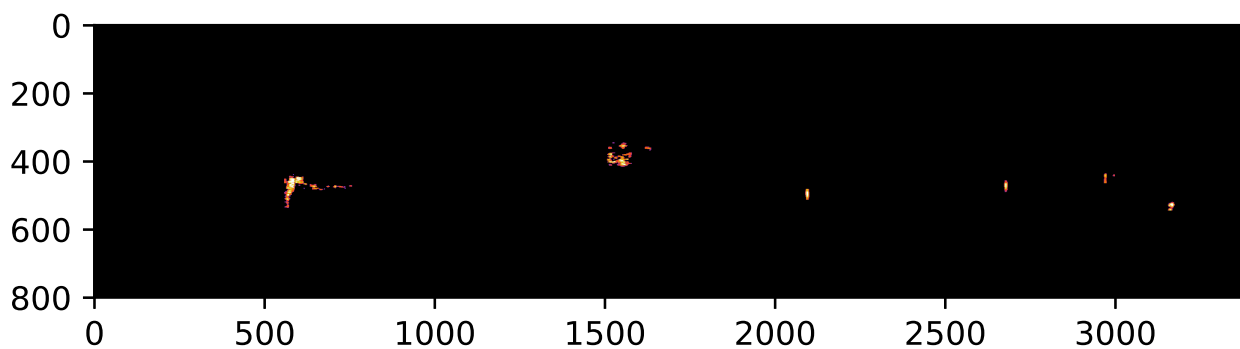


Figure 6 – Spectrogram after erosion, denoising, and dilation.

Erosion and dilation filters (Gonzalez & Woods, 2002) set each pixel to the minimum and maximum (respectively) over all pixels in a neighbourhood centred at the given pixel. Erosion filters shrink bright regions and enlarge dark regions, whilst dilation filters do the opposite. Erosion often breaks up regions and removes low- $\sigma$  noise (large signal-to-noise ratio), whilst dilation rejoins broken regions. A combination of the two, with a denoising filter in between, allows shapes to be smoothed, broken regions reconnected, and remaining noise removed (Figure 6).

A total-variation Chambolle denoising filter (Chambolle, 2004) is used due to its property of preserving edges whilst smoothing noise in flat regions, even with low signal-to-noise ratios.

### 3.5 Echo identification

Owing to the characteristic shape of meteor echoes as streaks, a kernel convolution (with kernel in Figure 7) replicating this shape is used to identify regions of the image that may contain echoes. A large kernel is used since echoes are large relative to the image dimensions (800 by 1133 for comparison data), on the order of 40 pixels in height and 10 pixels in width, at minimum. This is a common image processing technique, though is often only part of a more complex algorithm (Šustr, 2013). It is advantageous to use this method before region identification since artefacts may remain in the image after denoising, whether due to radio effects such as tropospheric propagation and interference, or artefacts of the prior processing, and these are often dissimilar to

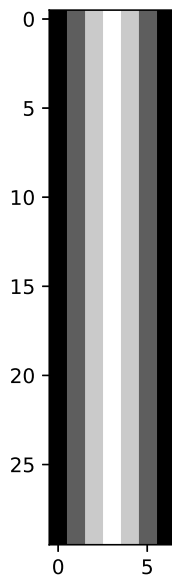


Figure 7 – Normalised kernel used for streak convolution.

the echoes in shape, so are easily removed (or at least nullified) with a streak convolution (see Figure 8).

Region identification is performed using connected-component analysis (CCA) (Dillencourt et al., 1992). This is a method of labelling image pixels based on connected components satisfying a given heuristic. Many algorithms exist and choice depends on convenience, availability for implementation in a given language, and speed. 4-connectivity is used in the interest of processing speed, since echo shapes are assumed to be basic and do not require 8-connectivity. Once regions are labelled, centres of mass are calculated with the R component of pixel RGB values as weights. This provides an approximate location for the centre of the echo. These centres are marked with crosses in the final results (Figure 5.1). This process gives Figure 9.

The dilation filter in the denoising process does not always reconnect broken regions, so sometimes echoes may still be split in two after this filter is applied. Similarly, the appearance of an echo in the spectrogram may simply be split on detection (see Figure 10). Thus, once images have been split into regions, these regions must be rejoined. This is done by identifying any region that overlaps horizontally with another region. These are then recategorised as the same region. Note that vertical overlapping is not considered since the time axis is horizontal, so horizontally distinct regions occur at distinct times and should not be considered the same echo. Despite this, some echoes occur so close in time that they overlap slightly. In order to separate these echoes the image is stretched horizontally, giving greater resolution in time.

In some spectrograms erroneous detections occur, primarily due to radio artefacts, and these can be identified from their vertical position: the majority of echoes are underdense which have their centre of mass close to the CW frequency, so by considering vertical locations outliers can be removed. The definition of an outlier as beyond  $1.5\times$  the inter-quartile range of the vertical

centre of mass locations is used, which is calculated on a per-image basis.

### 3.6 Calculations

In order to calculate the start and end time of a given echo, the horizontal location (in pixels) of the echo region bounding box is used, as well as a user-specified seconds-per-pixel value and the image's timestamp. Duration is then easily calculated from the start and end times, which are given in the timezone the original spectrogram uses (usually UTC).

The top and bottom locations of the echo region bounding box are used to estimate approach and recede velocities using the Doppler equation (1), which requires a user-specified Hz-per-pixel value and the demodulated CW frequency vertical location in the image, specified in pixels. Results are given in kilometres per second.

Maximum intensity is calculated by identifying the pixel in a given echo region with the greatest value relative to the colour scale in the image, which requires the user to locate the scale within the image during the pre-processing stage.

An overview of the design of the MESA (meteor echo spectrogram analysis) program is given in Figure 11.

## 4 Implementation details

The program is written in Python 3 (available at [www.python.org](http://www.python.org)).

### 4.1 Libraries

Libraries used are:

- NumPy (available at [www.numpy.org](http://www.numpy.org)) is a module used for computation throughout the program since it is optimised for vectorised data, so is efficient for working with images when represented as arrays.
- Matplotlib (available at [matplotlib.org](http://matplotlib.org)) is a plotting tool used to display the spectrogram and graphics associated with analysis, such as circling the estimated echo centres.
- OpenCV (available at [docs.opencv.org/3.0-beta/modules/core/doc/intro.html](http://docs.opencv.org/3.0-beta/modules/core/doc/intro.html)) is a computer vision module used for common image processing techniques such as denoising, erosion filters, and dilation filters.
- SciPy (available at [www.scipy.org](http://www.scipy.org)) is a scientific computing library used here for connected-component analysis.
- Scikit-Image (available at [scikit-image.org](http://scikit-image.org)) is an image processing library providing efficient implementations of Otsu thresholding and TV Chambolle denoising.

### 4.2 Pseudocode

Pseudocode for important processes in the program are shown in appendices B, C, and A, namely rejoining of broken echoes, estimation of max intensity for a given

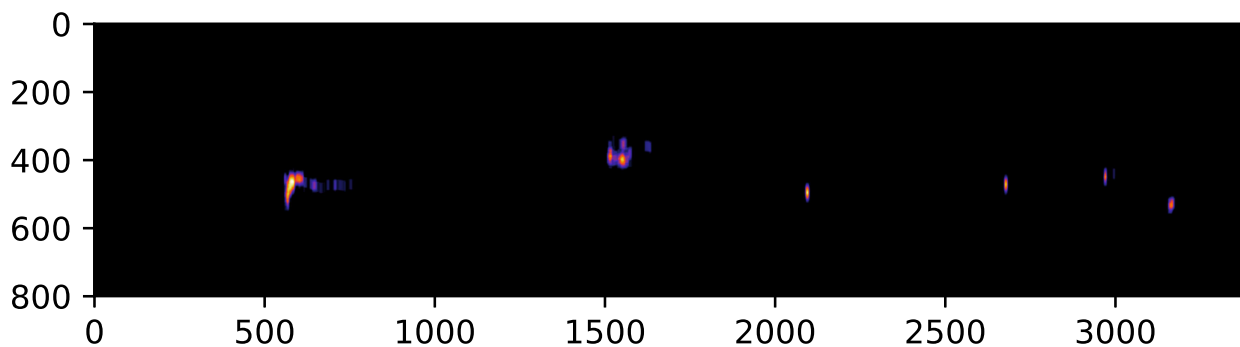


Figure 8 – Spectrogram after thresholding.

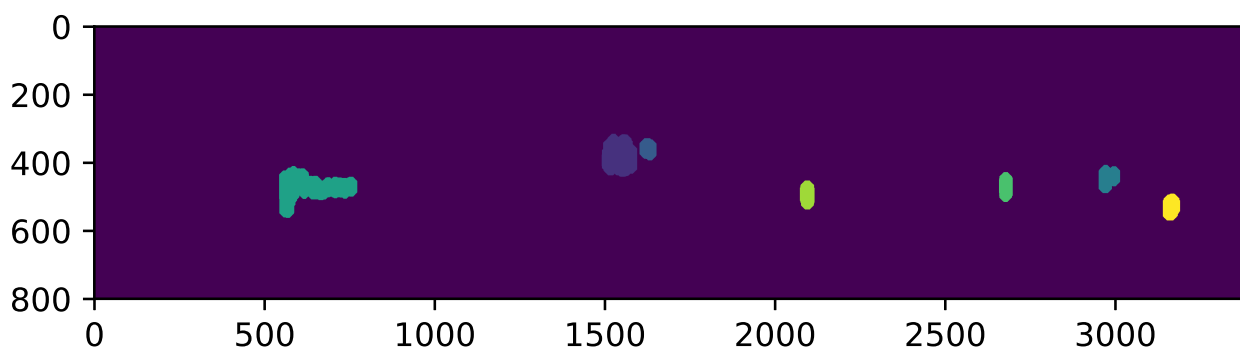


Figure 9 – Spectrogram after region labelling.

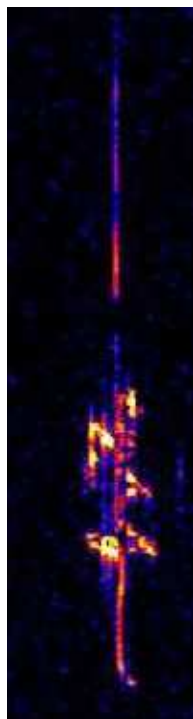


Figure 10 – Some echoes may be split vertically.

list of echo centre-of-masses, and removal of lines in pre-processing. Variables defined earlier in the program are underlined (and thus passed as an argument to the function if needed).

Algorithm 1 in appendix A shows the process for removing the vertical lines which indicate 30 second increments in the spectrogram. The mean RGB value of the bottom pixel of each line is checked and the greatest 9 are chosen (assuming there are 9 lines). The bottom pixel is checked since in the images the lowest pixel is where the bar starts, so is white, thus the mean RGB value of this pixel is approximately 255.

Algorithm 2 in appendix B shows the process for reconnecting broken echo regions. A vertical column strictly containing the given echo region is checked for other regions. Those identified are recategorised as part of the original region.

Algorithm 3 in appendix C is used to estimate the maximum intensity of a given detected echo. Each pixel in the region has its maximum intensity calculated: first the index is found for the part of the colour scale with the minimum colour difference, then linear interpolation between the corresponding intensity at the start and end of the scale gives the pixel's intensity. The maximum of these intensities and the corresponding pixel is then identified.

### 4.3 Calculations

Formulae for the various calculations are shown below using variables entered by the user (in **bold**), or calculated previously in the program (in *italics*).

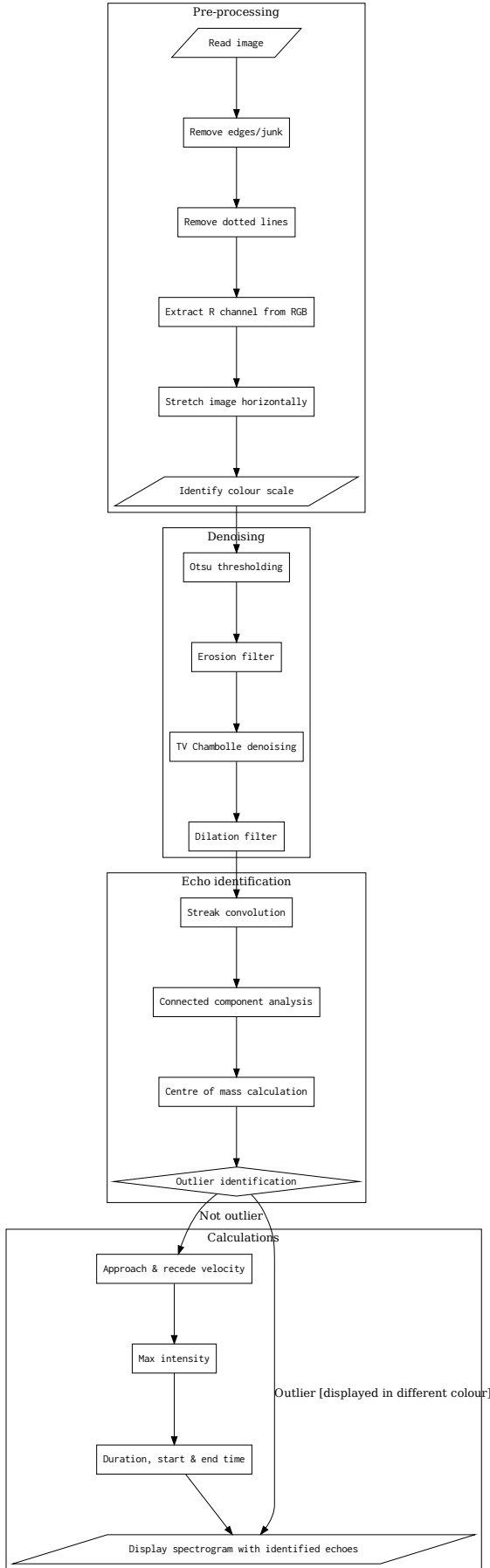


Figure 11 – Flowchart of MESA program.

### Approach & recede velocity

$$\begin{aligned}
 f_{\text{rec}} &= (\text{bottom} - \text{CWfreq}) \cdot \text{HzPerPixel} \\
 f_{\text{appr}} &= (\text{CWfreq} - \text{top}) \cdot \text{HzPerPixel} \\
 v_{\text{rec}} &= \frac{c \cdot f_{\text{rec}}}{f_{\text{CW}}} \\
 v_{\text{appr}} &= \frac{c \cdot f_{\text{appr}}}{f_{\text{CW}}}
 \end{aligned} \tag{2}$$

where *bottom* is the distance in pixels between the bottom of the echo region and the top edge of the image; *top* is defined similarly, **CWfreq** is the vertical location (in pixels) of the CW frequency in the spectrogram, **HzPerPixel** indicates Hz per vertical pixel, *c* is the speed of light in a vacuum, given in  $km \cdot s^{-1}$ . Results are given to an accuracy of 3dp.

### Max intensity

$$I_{\text{dB}} = \frac{i}{l} \cdot (s_{\text{min}} - s_{\text{max}}) - s_{\text{offset}} \tag{3}$$

where *i* is the index of the colour scale pixel with the minimum colour difference to the current pixel, and *l* is the length of the colour scale in pixels.  $s_{\text{max}}$ ,  $s_{\text{min}}$  are the maximum and minimum intensities on the scale when offset by  $s_{\text{offset}}$  so that  $s_{\text{min}}$  is 0. This variable, despite always being 0, is kept to aid clarity.

### Start & end time

$$\begin{aligned}
 \Delta t_{\text{start}} &= \text{sPerPixel} \cdot \left[ w_{\text{image}} - (\text{left} + n_{\text{lines}}^l) \right] \\
 t_{\text{start}} &= t_{\text{image}} - \Delta t_{\text{start}} \\
 \Delta t_{\text{end}} &= \text{sPerPixel} \cdot \left[ w_{\text{image}} - (\text{right} + n_{\text{lines}}^r) \right] \\
 t_{\text{end}} &= t_{\text{image}} - \Delta t_{\text{end}}
 \end{aligned} \tag{4}$$

where **sPerPixel** identifies the number of seconds per horizontal pixel,  $w_{\text{image}}$  is the width of the image, *right* is the distance in pixels of the right side of the echo region (where the left edge of the image is 0 and the right edge of the image is  $w_{\text{image}}$ ), *left* is defined similarly,  $n_{\text{lines}}^{l,r}$  is the number of lines removed between the left edge of the image and the left (or right) edge of the echo region, used to adjust the time based on how many pixels were removed in preprocessing.

## 5 Comparison

To test the performance of the MESA program, the sensitivity was calculated from a set of spectrograms analysed both manually and automatically. The equivalent of one day of data was analysed. Since each spectrogram covers 5 minutes, 288 images must be processed: first, every fifth spectrogram is selected from a week of data 15<sup>th</sup> August 2016 to 21<sup>st</sup> August 2016, inclusive, from which a random sample of 288 images was taken. A short timescale was used so that changes in the detection setup and spectrogram generation, such as the colour scale parameters, are unlikely to have occurred.

The performance of the MESA program was then analysed by manually processing the 288 images with



the program results overlaid and recording the number of true positives (successful detections), false positives (non-head echo detections), and false negatives (missed head echoes). This indicates the proportion of detections that are actually head echoes, and how often head echoes are missed. In order to test the performance of the program relative to how noisy the spectrogram is, the mean-squared error (MSE) of the pixel values was calculated as a measure of noise, so that the relationship with sensitivity can be seen. The sensitivity of the MESA program was then determined as the average proportion of true positives per spectrogram out of the true positives and false negatives for the same spectrogram. The sensitivity was then calculated for spectrograms with MSE below a given threshold and plotted against these thresholds, which ranged from 50 to 120. A histogram of the MSE noise is shown in Figure 12.

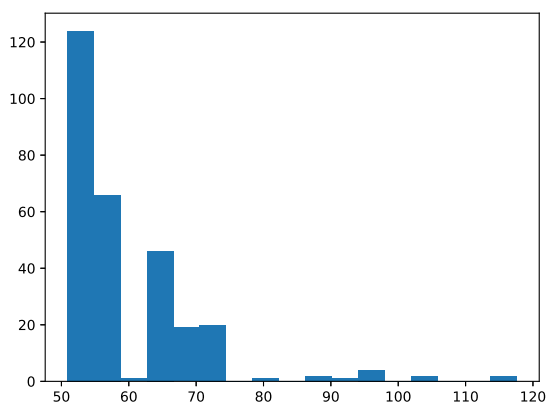


Figure 12 – Histogram of mean-squared error noise of the spectrogram data.

No analysis was made of the accuracy of regions identified around each head echo: the choice of bounding region is largely subjective and it would be almost impossible to quantify the performance of the program. The important aspect of the program is that it functions consistently, considering one of the purposes of automated analysis is to give an objective way to identify echo regions to allow comparison of different sets of data, whilst manual analyses by two different persons may disagree.

## 5.1 Results

**Performance analysis** In the 288 images there are in total 1048 successful detections, 1244 false detections, and 255 missed detections, resulting in a per-spectrogram average of 3.64 successful detections, 4.32 false detections, and 0.89 missed detections.

This shows that 54% of detections are not actually head echoes. However, 69% of spectrograms have no missed detections, and excluding blank spectrograms (that is, no successful, false, or missed detections) then 87% of spectrograms have no missed detections.

The sensitivity of the program has a clear inverse relationship with the MSE of the image being processed:

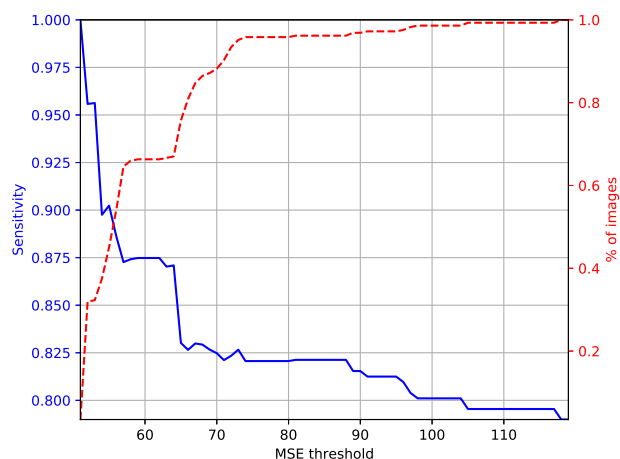


Figure 13 – Sensitivity against MSE threshold, with percentage of images with MSE less than the threshold.

see Figure 13. Including all spectrograms, the sensitivity of the program is 0.790. Including spectrograms only with MSE noise < 60, which accounts for  $\sim 65\%$  of the spectrograms, the sensitivity of the program rises to 0.875.

**Example spectrograms** Certain artefacts in spectrograms result in poor performance of the MESA program, for example high noise (Figure 14b), direct signals and aeroplanes (Figure 14c), and tropospheric propagation (Figure 14d). An ideal spectrogram is shown in Figure 14a.

## 6 Discussion

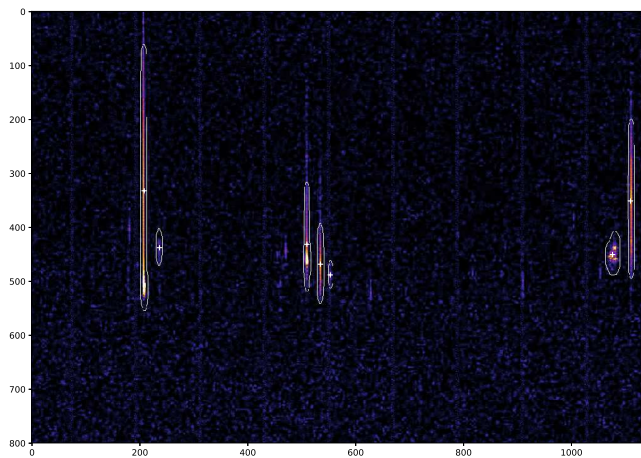
### 6.1 MESA performance

These results show that whilst a large proportion of the programs detections are not head echoes, relatively few head echoes are missed. The program performs well on spectrograms with low noise (MSE < 60): these are most commonly spectrograms without direct signal and other artefacts. Large amounts of noise (specifically signal-to-noise ratio) means the program cannot distinguish between echoes and noise. Thus different types of noise or artefacts impede the program in different ways. When the noise arises as ‘blobs’, the streak convolution removes this. If the noise arises as streaks, for example when aeroplanes have been detected (in which case there are effectively multiple very short streaks next to each other), the streak convolution doesn’t remove the noise and the program performs poorly. This also occurs when there is a direct signal. When the colour scale is not calibrated to eliminate noise, the denoising process is not capable of removing it.

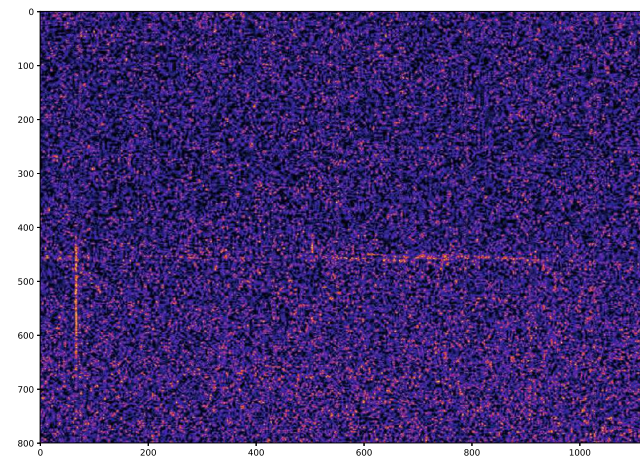
The results demonstrate that the MESA program does not perfectly replace manual processing, however it allows significant reduction in the volume of data that must be manually processed.

### 6.2 Limitations

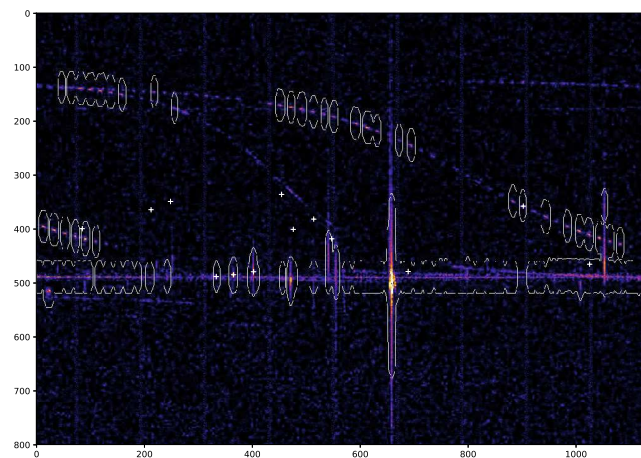
**Available data** Only 50% quality JPEG spectrograms were available for the analysis of MESA performance, whilst the program is envisaged for use with



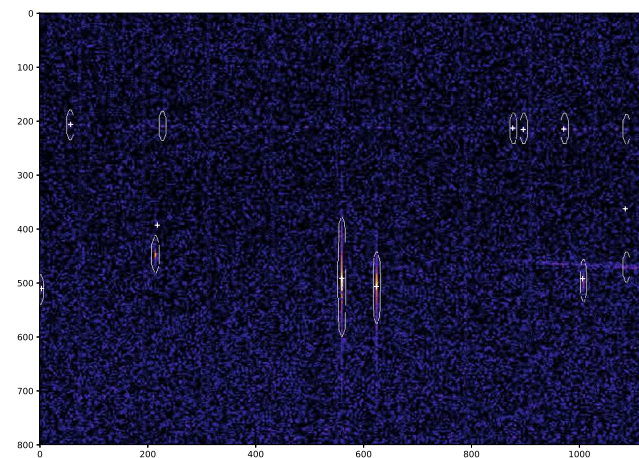
(a) The ideal spectrogram: the MESA program performs well.



(b) Spectrogram with high noise.



(c) Spectrogram containing artefacts from aeroplanes, and direct signal.



(d) Spectrogram containing tropospheric propagation.

Figure 14 – Examples of spectrograms and the performance of the MESA program on them.



higher-quality spectrograms, where the program will perform better, as demonstrated by the greater sensitivity at lower noise thresholds. Also, higher quality spectrograms will have fewer compression artefacts, which hinder performance. For example, the noise created by JPEG compression around the vertical time markers can cause false detections.

**Resolution** The resolution of the spectrogram image determines the resolution of the output data. Velocity data are displayed up to a resolution determined by the Hz-per-pixel value. Given a frequency resolution  $f_{\text{err}}$ , the velocity resolution is  $\frac{c \cdot f_{\text{err}}}{f_0}$ . The time resolution is determined by the seconds-per-pixel value.

**Velocity calculation** Calculation of velocity from a spectrogram is non-trivial and requires a detailed description of the detection geometry. In this paper we describe only a naïve method, which only gives accurate results for clearly defined head echoes that are simple structured, that is, conform to the expected ‘streak’ shape. However, identification of the head echo region may allow accurate calculation of velocity when more robust techniques and more detailed detection information available.

**Colour scale** Depending on the colour scale configuration, the greater intensity end of the scale may be saturated in the image (Figure 15) making the greater intensities indistinguishable by colour and consequently giving erroneous results.

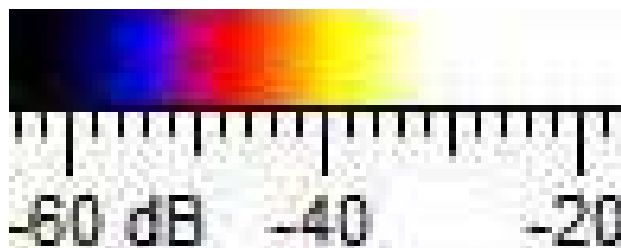


Figure 15 – CMRmap colour scale saturated at greater intensities

### 6.3 Advantages

**Pipeline integration** The MESA program can easily be integrated into a data pipeline, which offers an advantage over manual processing systems, even when many can contribute to the data analysis, for example the BRAMS Zooniverse project (Lamy et al., 2017). Data can be processed at a greater rate, with customisable settings, and there is less variation in analysis than when completed by multiple people.

**Reliability** Using an automated program ensures reliability when analysing multiple spectrograms, and even across multiple detection stations. This is extremely important when comparing data from multiple sources, and cannot always be guaranteed with other methods, especially manual analysis.

**Historical data** Detection stations often record data only in spectrogram format, since this requires far less memory storage than archiving the original radio signal. This backlog of data can be processed using MESA to give a large archive of detection counts and other data.

### 6.4 Future work

**Adjustable sensitivity** Most stages of the MESA program have adjustable settings: horizontal stretching factor (which is part of determining the time resolution), thresholding sensitivity, erosion and dilation strength, and CCA structural elements, TV Chambolle denoising weight, streak convolution kernel, and the outlier identification process. Combinations of certain parameters could be tested to give a powerful tool to detect certain elements of spectrograms: overdense or underdense meteors, for example. Thresholds could be set to only detect meteors above a given minimum intensity, or beyond a minimum duration.

**Acceleration calculation** Provided higher resolution spectrograms are available, the acceleration of an echo-producing meteor could be estimated by identifying the points of maximum and minimum velocity, and determining the time-separation of these points.

## 7 Conclusion

- Meteor head echoes may be automatically identified in spectrograms generated by forward-scatter radio detection, as well as bounding regions which allow calculation of further data including velocity and duration.
- Although the program does not perfectly replace manual processing, it allows significant reduction in the volume of data that must be manually processed.
- The MESA program allows large volumes of spectrogram data to be processed at a sensitivity of 0.790, with a greater sensitivity when low-noise data is processed: a sensitivity of 0.875 is attained with  $\sim 65\%$  of the data below the noise threshold.
- This allows a consistent method of automatically identifying head echoes and the region that defines the echo.
- The program is easily extensible to allow automatic calculations relating to the echoes.
- Each step of the program is adjustable, so that the program can be customised for a given detection setup, or targeted at certain types of echoes.

## Acknowledgements

This work has been funded by the Devon Cambridge Society. I am grateful to the Norman Lockyer Observatory for access to spectrogram data, and Michael Andrejczuk for suggestions and advice.

## References

- Chambolle A. (2004). “An algorithm for total variation minimisation and applications”. *Journal of Mathematical Imaging and Vision*, **20:1-2**, 89–97.
- Dillencourt M., Samet H., and Tamminen M. (1992). “A general approach to connected-component labeling for arbitrary image representations”. *Journal of the ACM*, **39:2**, 253–280.
- Gonzalez R. C. and Woods R. E. (2002). *Digital image processing*. Prentise Hall, New Jersey, 2 edition.
- Lamy H., Calders S., Tétard C., Verbeeck C., Martinez Picar A., and Gamby E. (2017). “The Radio Meteor Zoo: searching for meteors in BRAMS radio observations”. *European Planetary Science Congress*, **11**, EPSC2017–714.
- Noguchi K. and Yamamoto M. (2008). “Development of an automatic echo-counting program for HROFFT spectrograms”. *Earth, Moon, and Planets*, **102**, 323–329.
- Otsu N. (1979). “A threshold selection method from gray-level histograms”. *IEEE Trans. Sys., Man., Cyber.*, **9**, 62–66.
- Roman V. and Buiu C. (2015). “Automatic detection of meteors in spectrograms using artificial neural networks”. In *10th Jubilee IEEE Symposium on Applied Computational Intelligence and Informatics*. pages 131–134.
- Šustr M. (2013). “Streak detection in astronomical images”. Bachelor thesis, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, 38 pages.

---

*Handling Editor:* Jean-Louis Rault

This paper has been typeset from a  $\text{\LaTeX}$  file prepared by the author.

## Appendix

### A Time-stamp indicator removal

---

**Algorithm 1** Vertical timestamp-indicator line removal in pre-processing

---

- 1:  $\underline{img} \leftarrow$  spectrogram after border removal
  - 2:  $\underline{indices} \leftarrow$  indices of columns with largest mean RGB pixel value in bottom pixel  $\triangleright$  Ordered from largest to smallest
  - 3: **initialise**  $\underline{lineLocations}$
  - 4:  $\underline{imgNew} \leftarrow$  deep copy of  $\underline{img}$
  - 5: **for**  $i$  between 0, (length of  $\underline{indices}$ ) – 1 **do**
  - 6:      $\underline{ind} \leftarrow$  column index at index  $i$  in  $\underline{indices}$
  - 7:      $\underline{imgNew} \leftarrow \underline{img}$  with column  $\underline{ind}$  removed
  - 8:      $\underline{lineLocations} \leftarrow \underline{ind} - i$
  - 9: **end for**
-

## B Broken region reconnection

---

### Algorithm 2 Reconnection of broken echo regions

---

```

1:  $lbls \leftarrow$  array of region labels with same dimensions as spectrogram after pre-processing
2: for all region labels do
3:    $com \leftarrow$  CoM of current region
4:   if  $com$  is a valid co-ordinate then
5:      $x \leftarrow$  x co-ordinate
6:      $y \leftarrow$  y co-ordinate
7:      $rgnVal \leftarrow$  label of current region
8:     while region label at  $(x, y) = rgnVal$  and  $x <$  image width do
9:       increment  $x$ 
10:    end while
11:     $rgnEnd \leftarrow x$ 
12:     $rgnStart \leftarrow$  index of  $rgnVal$  in array of row  $y$ 
13:     $column \leftarrow$  columns from  $x = rgnStart$  to  $x = rgnEnd$ 
14:     $otherRgns \leftarrow$  list of region labels (excluding  $rgnVal$  & 0) in the columns from  $rgnStart$  to  $rgnEnd$ 
15:    for  $labelVal$  in  $otherRgns$  do
16:      replace all elements in  $lbls$  with region label  $labelVal$  with  $rgnVal$ 
17:    end for
18:  end if
19: end for

```

---

## C Maximum intensity calculation

---

### Algorithm 3 Maximum intensity calculation

---

```

1: for  $i$  between 1, length of list of echo centres of mass do
2:    $coords \leftarrow$  centre of mass for region with label  $i$ 
3:    $dims \leftarrow$  dimensions of region with label  $i$ 
4:    $scaleStart \leftarrow 0$ 
5:    $scaleEnd \leftarrow$  length in pixels of colour scale in image
6:   if  $coms$  is a valid co-ordinate then
7:      $lbls \leftarrow$  array of region labels with same dimensions as image
8:      $mask \leftarrow$  binary array where  $(x, y)$  in  $mask$  is 1 if  $(x, y)$  in  $lbls$  has value  $i$ , and 0 otherwise
9:      $img\_lbl\_rgb \leftarrow$  copy of pre-processed image (after stretching, red channel extraction, line removal)
10:    replace elements of  $img\_lbl\_rgb$  corresponding to zeros in  $mask$  with 0
11:    initialise  $scale\_vals$  as empty list
12:    for  $(row, col)$  in  $img\_lbl\_rgb$  do
13:       $pxlVal \leftarrow$   $img\_lbl\_rgb$  value at  $(row, col)$ 
14:      if  $pxlVal$  non-zero then
15:        initialise  $comparisons$  as empty list
16:        for  $scaleImgVal$  in array of image values of scale do
17:           $comp \leftarrow |\mathbf{mean}(scaleImgVal - pxlVal)|$   $\triangleright$  Note subtraction is channel-wise, then mean is
over the three channels
18:          append  $comp$  to  $comparisons$ 
19:        end for
20:         $minIndex \leftarrow$  index of minimum value in  $comparisons$ 
21:         $scaleVal \leftarrow \left[ \frac{minIndex}{\text{length of } comparisons} \times (scaleStart - scaleEnd) \right] - offset$ 
22:      end if
23:    end for
24:  end if
25: end for

```

---