Exploration of Signal Reconstruction and Time-Scale Modification

By: Allison Crim

Introduction

In audio signal processing applications, it is important to be able to reconstruct a signal without having access to its phase information. The first part of this report explores a method of signal reconstruction described in the paper 'Real-Time Signal Estimation From Modified Short-Time Fourier Transform Magnitude Spectra' by Zhu et al which results in a near artifact-free signal. The method presented in the paper also can be applied to perform time-scale modification on a signal. The second part of this report compares the effectiveness of time scale modification using the RTISI method presented by Zhu et al and the overlap-add method presented by Dreidger and Muller.

Signal Reconstruction using RTISI

The method of signal reconstruction presented in the Zhu et al paper is called real-time iterative spectrogram inversion (RTISI). This algorithm is an alternative to a previously proposed algorithm for signal reconstruction called the Griffin and Lim (G&L) algorithm. RTISI differs from the G&L algorithm by processing the data in a frame-by-frame fashion instead of all at once. This allows for the algorithm to be implemented in real-time by taking the G&L algorithm and doing it piece by piece. The RTISI algorithm was implemented in matlab using the following steps.

To begin, a partial frame for the signal must be determined. The first partial frame of the signal will comprise of all zeroes because there are no previous samples to overlap add in. The next partial frames will be made using contributions from previous frames. This will be discussed further in the report.

The next step in the algorithm is to define the current frame in order to calculate the magnitude needed for estimating frame *m.* The current frame is determined by several parameters. The parameter *L* is defined as the frame length which is the length that each chunk of the signal is when broken up. Each frame is separated by the parameter *S* which is defined as the step size and in this paper, $S = L/4$ "..so that the *m*th partial frame comes from the overlap-added results of the…" previous 3 frames. Changing the value of *S* and its effect on results will be explored further on in this report.

After the frame is defined, it is windowed and an fft is taken to get the magnitude of the signal in the frequency domain. The window used in this algorithm is a scaled Hamming window whose equation is shown below. In this paper, $a = 0.54$ and $b = -0.46$.

$$w(n) = \begin{cases} \frac{2\sqrt{S}}{\sqrt{(4a^2+2b^2)L}}\left(a + bcos\left(2\pi\frac{n}{L}\right)\right), & if\ 1 \leq n \leq L \\ 0, & otherwise \end{cases} \qquad eq.1$$

After the magnitude is found, the iteration step begins to estimate the phase and come up with a final value for *m*. To begin the iteration, the algorithm requires windowing and taking the fft of the partial frame. Following this calculation, the phase estimation can be completed by using the *angle()* function in matlab. Next, to find the i^{th} estimation of frame *m,* the magnitude and phase calculated above are combined and the inverse fft is taken. This process is repeated *I* number of times to find the final frame *m*. The impact of changing the value of *i* will be explored later in the report.

The final step in the algorithm is to overlap add the final frame *m* with the partial frame and then move on to the next frame. When the frame is 'committed', it is then used to create the next partial frame. This partial frame is created by combining the final frame *m* and the previous partial frame. This is iterated through until the entire signal has been processed.

Result of Varying Input Parameters

In this section, the effect of varying input parameters is explored. In order to quantify the performance of the RTISI algorithm, a calculation of signal-to-error ratio (SER) was performed. The equation for calculating SER is shown below.

$$SER = \frac{\sum_{m=-\infty}^{\infty}\frac{1}{2\pi}\int|X(mS,w|^{\,2}dw}{\sum_{m=-\infty}^{\infty}\frac{1}{2\pi}\int|X(mS,w|-|X'(mS,w)|^2dw} \qquad eq.2$$

In the RTISI algorithm, several parameters can be varied to improve the performance. The parameters that were explored in this project were number of iterations, step size, and frame size. The number of iterations is explored in the table below. The increased number of iterations implies an increased computational requirement so the user must balance the desire for fewer computations vs performance. It is clear that as the number of iterations increases, the performance improves.

Table 1: Evaluation of Changing Number of Iterations

Number of Iterations	SER
2	27.67
8	28.87
80	29.13

The next parameter that was explored is the step size. In this table the values of 87.5%, 75% and 50% were compared. In this project, no asymmetric windowing was implemented so the larger step sizes increased performance. Having a smaller step size requires more computation so choosing a larger step size is beneficial.

Table 2: Evaluation of Changing Step Size

Step Size	SER
L/8	25.31
L/4	28.87
L/2	31.11

The final parameter explored in this project is frame size. The best result came from a frame size of 1024. Values greater and less than 1024 resulted in declined performance. Because computational requirements decline with a larger frame size, it is important to choose the proper frame size to balance that with performance.

Table 3: Evaluation of Changing Frame Size

Frame Size	SER
256	28.42
512	28.87
1024	30.55

RTISI worked significantly better for vocal signals than music signals based on their SER values. This may be because the music signals were significantly more complex than the vocal signals.

Time-Scale Modification Using RTISI

The RTISI algorithm explored above can be used to perform time-scale modification. Time-scale modification is the process of speeding up or slowing down a signal without affecting the pitch. In order to perform time-scale modification with RTISI, the parameter previously known as S (step size) is now defined as S_s which is now the synthesis stepsize. Another parameter, S_a is introduced and it is used to do the STFTM. The paper defined S_a as S_s/α, with α representing the modification rate. When the modification rate is greater than 1, the signal sounds sped up and when the modification rate is less than 1 the signal sounds slowed down. According to the Zhu et al paper, choosing a value for α that results in a value of $S_a < L$ will compromise the effectiveness of the TSM because there will be no overlap between the frames.

Time-Scale Modification Using OLA

Another method available for time scale modification is overlap-add. In the paper "A Review of Time-Scale Modification of Music Signals" by Dreidger and Muller, the algorithm for implementing the overlap-add is described. This method is computed in the time domain which causes it to be better suited for processing percussive signals than harmonic signals. To perform this operation, start with separating the signal into analysis frames of length N spaced by analysis hopsize H_a. Next, the analysis frame is windowed using a Hann window and the windowed frames are relocated on the time axis a distance of H_s. Finally, all of the frames are added together to generate the output signal.

Comparison of TSM Implementations

Because there was no quantitative method for measuring the effectiveness of the time-scale modification presented in the paper, a subjective approach was taken for evaluation. Listening to the signals for any distortion and or strangeness provided a way to rate the effectiveness of each method. It is clear that the OLA version sounded better than the RTISI TSM for vocal and music signals. In the RTISI version there seems to be a weird effect on the background sounds that does not seem to be as present in the OLA version. The clear tradeoff that exists between the two is that the OLA version took a significantly longer amount of time to process the data. When the signals were long enough, OLA could not process them in a reasonable amount of time. The OLA method presented in the Dreidger and Muller paper was significantly easier to implement than the RTISI method. This is a clear example of the age-old tradeoff between computational efficiency, ease of implementation, and performance.

Conclusion

An exploration of the impact of varying parameters in the RTISI algorithm was completed based off of the algorithm presented in the Zhu et al paper. This resulted in a better signal when the number of iterations is increased, the step size is increased, and the frame size is chosen to be around 1024. These particular choices create a higher computational load but it is up to the user of the algorithm to decide if that satisfies their particular goals.

Extending the usefulness of the RTISI algorithm further, another experiment was conducted to compare the effectiveness of an RTISI based time-scale modification with an OLA based time-scale modification. In tests using the human ear as the judge, the OLA based TSM performed significantly better but has the tradeoff of not working in a time efficient manner for longer signals. Once again, it is up to the user to decide which is more important in their application; performance or efficiency.

References

Driedger, J., & Müller, M. (2016). A review of time-scale modification of music signals. *Applied Sciences*, *6*(2), 57. https://doi.org/10.3390/app6020057

Zhu, X., Beauregard, G. T., & Wyse, L. L. (2007). Real-time signal estimation from modified short-time Fourier transform magnitude spectra. *IEEE Transactions on Audio, Speech and Language Processing*, *15*(5), 1645–1653. https://doi.org/10.1109/tasl.2007.899236