Acoustics 2021
Wollongong, Australia
MAKING WAVES

Application of Improved Sliding DFT Algorithm for Non-Integer k

Carl Q. Howard

School of Mechanical Engineering, The University of Adelaide, South Australia, Australia

ABSTRACT

An algorithm and network is described in this paper that implements a sliding Discrete Fourier Transform, such that it outputs an estimate of the DFT value for every input sample. Regular DFT algorithms calculate a complex value that is proportional to the amplitude and phase of an equivalent sine wave at the selected analysis frequency. The analysis frequency that can be selected is typically an integer multiple of the frequency increment of the DFT algorithm, and this might not necessarily correspond to the desired analysis frequency. The sliding DFT algorithm proposed here overcomes this limitation, and permits the analysis frequency to be any value up to half the sampling frequency. The proposed sliding DFT algorithm is demonstrated by analysing a synthetic sine wave, and the exhaust noise from a V8 diesel engine.

1 INTRODUCTION

The Fast Fourier Transform FFT is a computationally efficient method of analysing a complicated signal to determine the amplitudes of equivalent sine waves over a range of frequencies. However, for situations where the amplitude at a single, or a few, frequencies are desired, other calculation methods are available that are more computationally efficient. One such algorithm is the Discrete Fourier Transform DFT that outputs a complex number

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi(kn/N)} \tag{1}$$

that encapsulates the amplitude and phase of an equivalent sine wave at analysis frequency kf_s/N Hz, for an input signal x_n that is sampled at frequency f_s Hz, at sample number $n = 0,1,..N-1$. The integer value N is the number of frequency 'bins' in the traditional FFT algorithm, and the integer value k is the bin number.

There are a couple of practical issues with implementing Eq. (1) on a digital signal processor to occur in real-time. First, the output value of $X(k)$ is only available after N samples have been processed, and multiples thereafter. While the calculation is being undertaken, no estimate of the current amplitude of the equivalent sine wave is available. An improvement to this limitation is the sliding-DFT algorithm (Jacobsen and Lyons (2003), Jacobsen and Lyons (2004), Duda (2010)) that outputs an estimate of $X(k)$ at every time sample. A further improvement is the sliding-Goertzel algorithm (Chicharo and Kilani (1996)) which is computationally efficient, and outputs an estimate of the DFT at each time sample.

However, most of the previous algorithms require that the selected analysis frequency must correspond to integer values of k and N, resulting in the analysis frequency to determine the amplitude and phase of the equivalent sine wave of kf_s/N Hz. It would beneficial if the analysis frequency that was selected was precisely the frequency of interest, instead of selecting the nearest frequency "bin" that encapsulates it. Sysel and Rajmic (2012) proposed a slight improvement to the sliding-DFT and sliding-Goertzel algorithm, where non-integer values of k could be used, enabling the selection of the exact analysis frequency, but the drawback remained that the output of the estimate of the DFT only occurs after N samples, and as it does not have a 'sliding' feature, the inputs must be 'reset' after N samples.

2 PROPOSED ALGORITHM

Lyons and Howard (2021) proposed an algorithm and network, as shown in Figure 1, that addressed both of these issues, namely, the ability to precisely select the analysis frequency by permitting non-integer values of k, and a 'sliding' feature so that an estimate of the DFT is output at every sample.

One application for this algorithm is in an automatic control system, where it is desirable to have a fast assessment of the effect of adjustments made to a feedback or feedfoward system, which enables the rapid control (usually minimisation) of some parameter, such as noise or vibration at a particular frequency. For example, in an adaptive-

passive control system, often it is not necessary to know the exact value of the 'cost-function', which is the parameter that is being minimised, but rather it is sufficient to know whether the previous adjustment to the system reduced or increased the value of the cost-function. This would enable the next update to the control system and adjustment to the actuators to be made, thereby achieving more rapid control, compared with using an algorithm that is only able to provide an estimate of the cost-function after *N* samples.

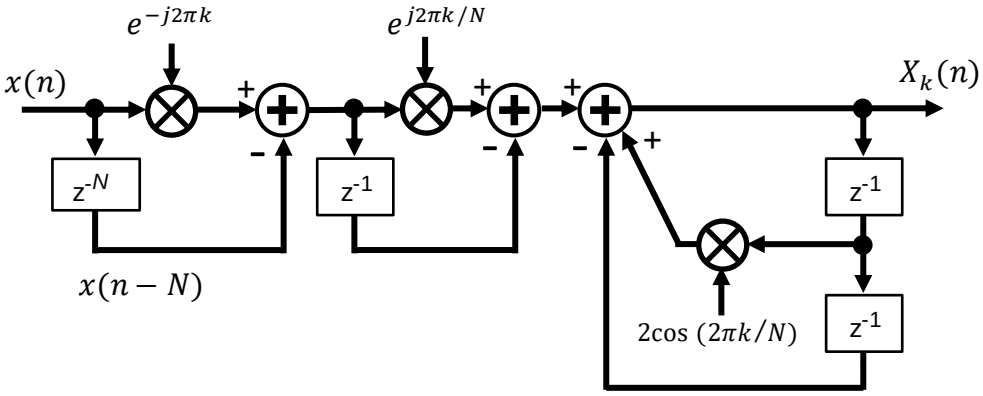

Figure 1: Network of the sliding-DFT algorithm for non-integer k (Lyons and Howard (2021), Fig 5).

The network shown in Figure 1 can be implemented in Matlab-Simulink, as shown in Figure 2. The Matlab-Simulink model is available at Howard (2021) on the Mathworks File Exchange repository. The input ports 1 and 2 are the input signal sampled at a frequency of Fs, and the value of k, respectively. The outputs ports 1 and 2 are the amplitude and phase of the calculated DFT value, respectively. A common misconception is that the calculated DFT value is the amplitude of the equivalent sine wave, which it is not, and needs to be rescaled by multiplying the calculated DFT value by 2/nDFT. For this Simulink model, the rescaling of the DFT is not undertaken to enable comparison with alternative DFT algorithms. The 'Integer Delay' block is shown as z^{-d}, where d = N = nDFT, which is a First-In-First-Out buffer of N samples. The triangular blocks are gain (multiplier) operators, the circular blocks are summation and subtraction operators, and the square and rectangular blocks are mathematical operators.

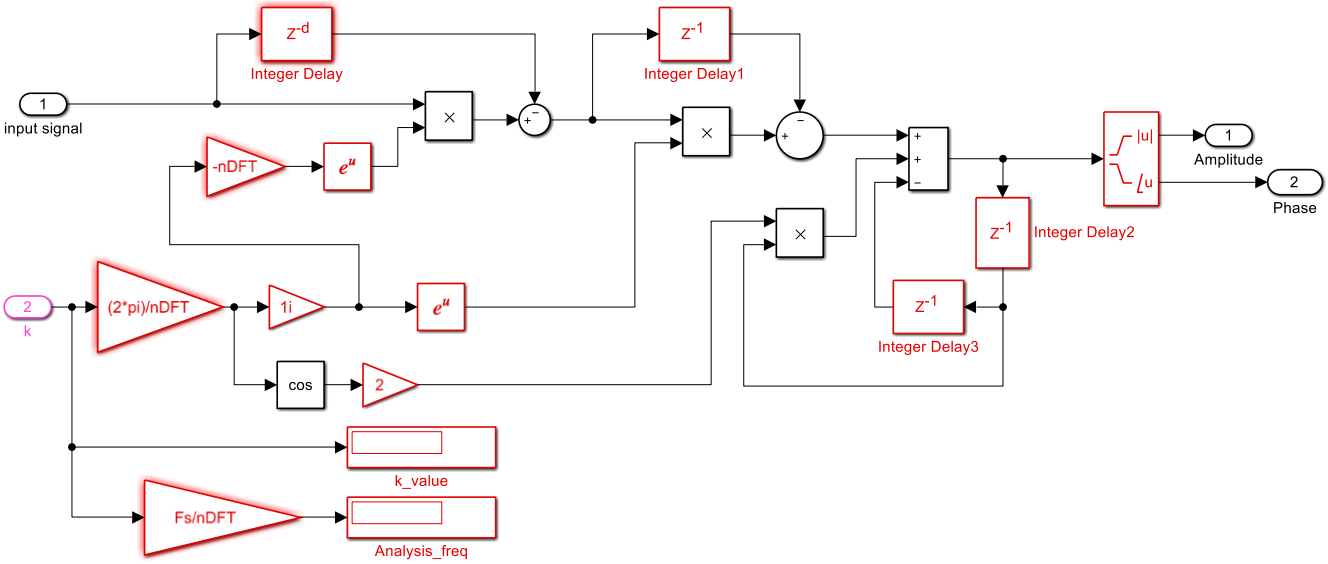

Figure 2: Matlab-Simulink model of the sliding-DFT algorithm for non-integer k.

3 APPLICATIONS

This section shows the application of the proposed sliding-DFT algorithm to a synthetic sine wave signal that varies in amplitude and phase over time, compared with algorithms proposed by other researchers. First, a comparison is shown where the excitation and analysis frequencies correspond to an integer value of k. The results

show that most of the algorithms are able to calculate the DFT amplitude correctly, which is to be expected. Next, the excitation and analysis frequencies are altered to correspond to a non-integer value of k. The results show that most of the algorithms are unable to calculate the DFT correctly, whereas the proposed algorithm, shown in Figure 1 and Figure 2, performs well.

Following the application of the proposed algorithm to a synthetic sine wave signal, the algorithm is used to analyse the exhaust noise from a V8 turbo-charged diesel engine to extract the amplitude of the tonal noise at the cylinder firing frequency. Previous work by Howard and Craig (2014a,b) involved the use of an adaptive-passive quarter-wavelength tube to attenuate a tone in the engine exhaust noise. A sliding-Goertzel algorithm was used to extract the amplitude of the tone, which can only occur for integer values of k. As the analysis frequency kf_s/N Hz usually did not correspond precisely to the cylinder firing frequency, it was necessary to employ additional signal processing techniques, such as summation of DFT values from nearby 'bins', and using frequency domain windowing of the input signal. Data from the experiments conducted in 2014 was re-analysed here using the proposed SDFT algorithm, where the use of a non-integer value of k was possible using the proposed algorithm in Figure 1 and Figure 2, and the results suggest that the proposed algorithm is able to rapidly track the amplitude of the tonal noise.

3.1 Varying Sine Wave

To compare the behavior of the DFT algorithms, simulations using Matlab-Simulink models were conducted. A sine wave that varied in amplitude between 1 and 2, that had a frequency of 15.0 Hz, and an initial phase angle offset of 45 degrees, was modelled in Simulink as shown in Figure 3. To enable comparison with output of the various DFT algorithms, the amplitude of the sine wave was multiplied by nDFT/2, where for this example nDFT=500.

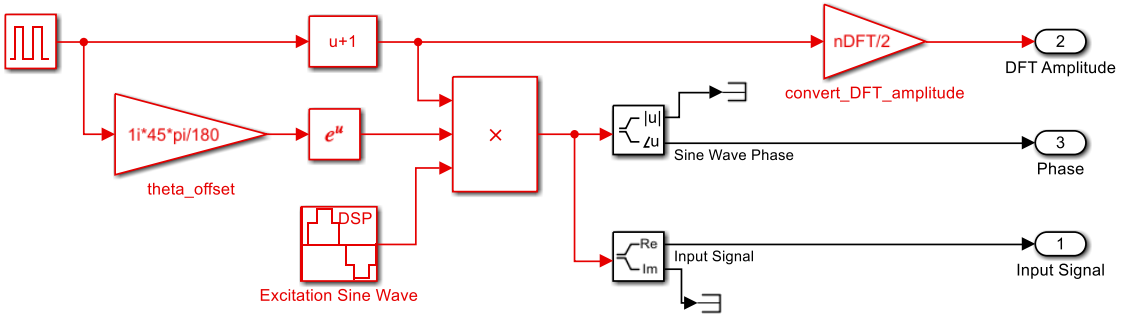

Figure 3: Simulink model of varying sine wave excitation source.

Figure 4 shows the network proposed by Duda (2010, Fig 4), which is a modulating mDFT network that has a sliding capability such that it will output a value of DFT for each input sample, and is a stable network when k is an integer.

Figure 5 shows a network proposed by Sysel and Rajmic (2012, Fig 2), that provides an estimate of the DFT amplitude when k is an integer or non-integer value. However, the network does not have a sliding capability and the calculation method would need to be restarted after every nDFT samples.

Figure 6 shows a Simulink model used to calculate the FFT of an input signal. An overlap of 75% is achieved using a first-in-first-out buffer, thereby reducing the time to output an estimate of the FFT amplitude. A Hanning window is applied to the input signal, the FFT magnitudes for all nFFT= 2^9 = 512 points are calculated, and then a subset of the appropriate FFT bin corresponding to the excitation frequency and the adjacent bins are summed to output the FFT magnitude of the input signal.

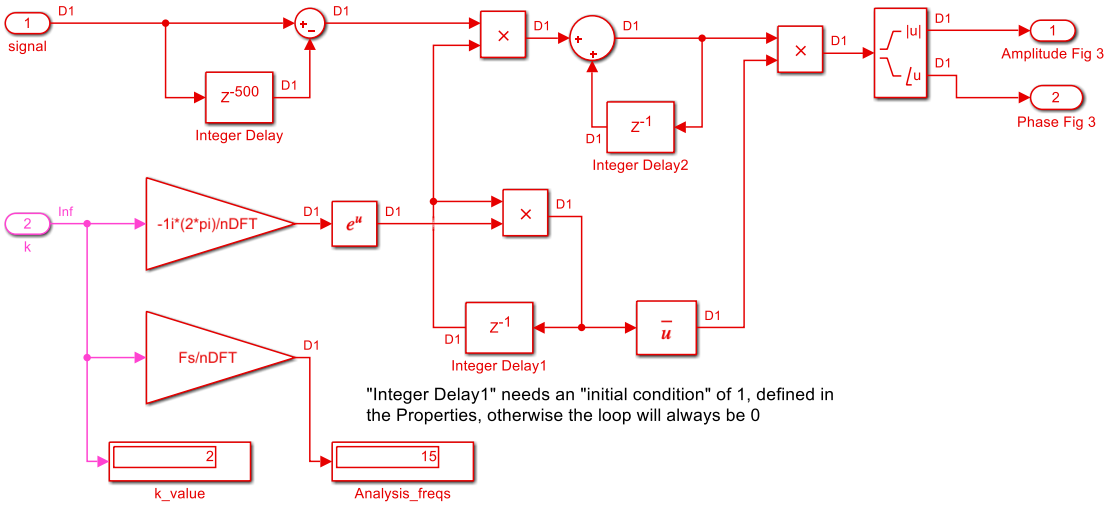

Figure 4: Modulating mDFT network proposed by Duda (2010, Fig 4.)

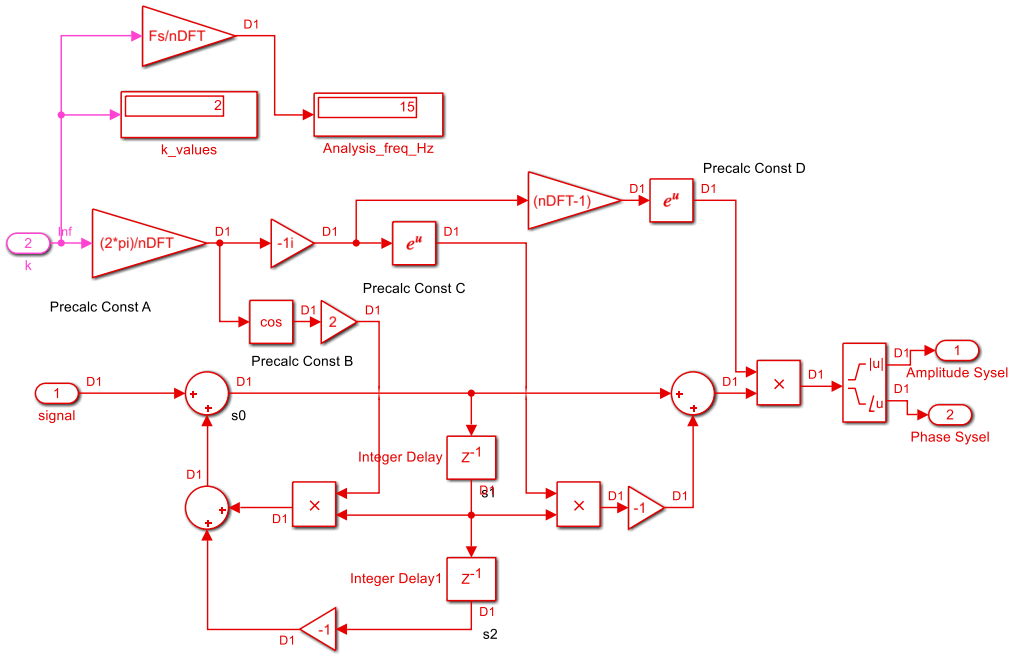

Figure 5: Network proposed by Sysel and Rajmic (2012, Fig 2), for non-integer k.

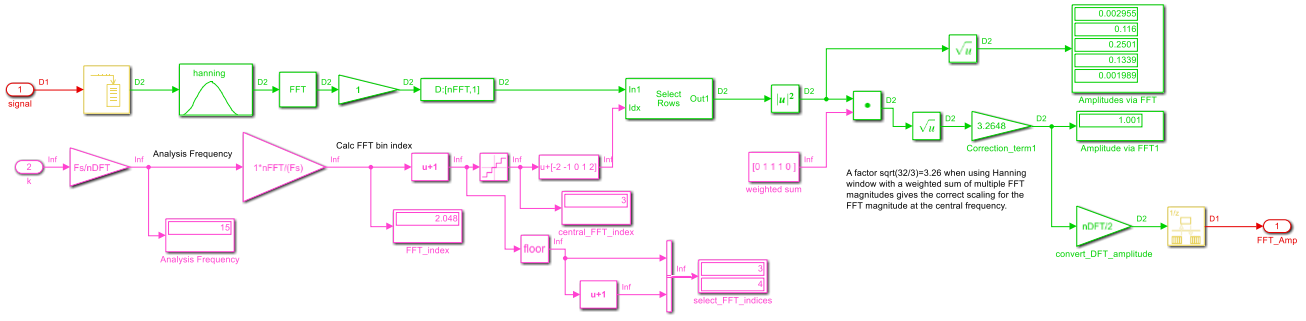

Figure 6: Simulink model used to calculate the FFT of an input signal, that utilizes the sum of the central FFT bin and the adjacent bins.

The DFT amplitudes of the input varying sine wave, shown in Figure 3, were calculated using the

- proposed network from Lyons and Howard (2021, Fig 5), shown here in Figure 2,
- modulating mDFT network proposed by Duda (2010, Fig 4), shown here in Figure 4,
- the network proposed by Sysel and Rajmic (2012, Fig 2), for non-integer k, shown here in Figure 5, and
- a traditional method using an FFT algorithm, shown here in Figure 6.

Initially these algorithms were compared using a sine wave of frequency 15.0 Hz, a sampling rate of Fs=3750 Hz, nDFT=500 samples, resulting in a value of $k = 15 \times 500/3750 = 2.0$, an integer value, as shown in Figure 7. The black solid line shows the DFT amplitude of the input sine wave signal, where the amplitude varies every 1.0s, and can be considered the desired output value that the other algorithms should calculate. The thick gray line shows the DFT estimate using the algorithm proposed by Duda (2010) and shows it tracks rapidly to the desired DFT value (black line). The light blue dotted line shows the estimate using the Sysel and Ramjic network and shows that it initially calculates the correct value after nDFT = 500 samples = 500/3750 = 0.13s, however it continues to diverge, as it does not have a sliding capability and it would be necessary to restart the algorithm every nDFT samples. The blue dash-dot line shows a traditional FFT based method and shows that it tracks the desired DFT value (black line) when there are changes in the input signal. The dashed yellow line show the DFT estimate using the Lyons and Howard (2021) algorithm, and shows that it has the same output as the thick gray line of the network proposed by Duda (2010).

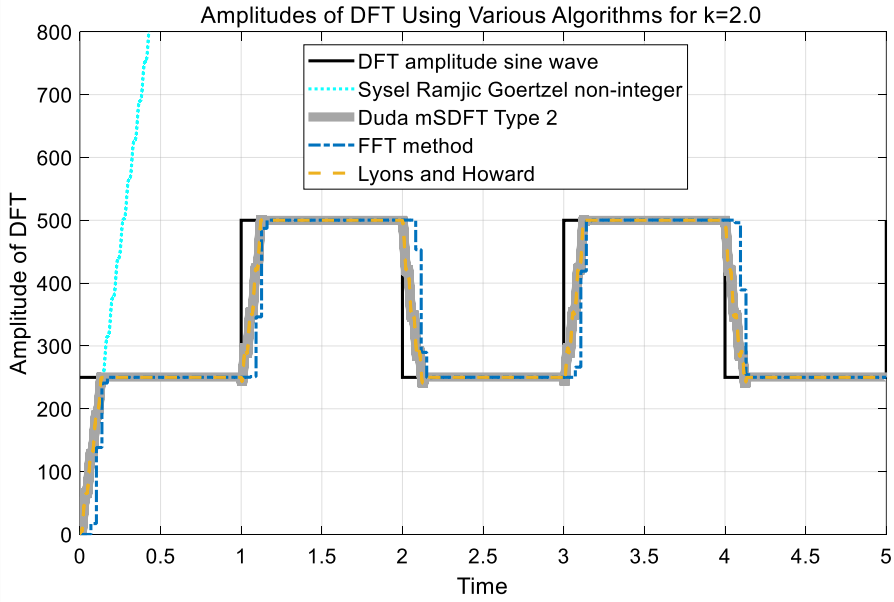

Figure 7: Comparison of DFT amplitudes calculated using various algorithms for k=2.0.

Figure 8 shows a close-up view of Figure 7 where the time axis has been zoomed to 0.8s to 2.2s to highlight the convergence of the algorithms. At time 1.0s, the amplitude of the 15.0 Hz sine wave was altered from 1.0 to 2.0, resulting in a transition of DFT amplitude (black line) from 250 to 500. It can be seen that the algorithms by Duda (2010) (gray line), and the Lyons and Howard (2021) (yellow dashed line) start to transition immediately. However, the estimate using the FFT method (blue dash-dot line) only starts to transition towards the correct value after about 0.1s.

The ability to start trending towards the correct DFT amplitude is important when used in a control system, as it enables the rapid convergence of the control system, such as minimizing noise in an active noise control system. It is not necessary to know the precise estimate of the amplitude of the cost function, which would be the amplitude of the sine wave in this example, but merely an indication of whether a previous adjustment of the controller output had resulted in an increase or decrease of the cost-function. In this example, the correct estimate of the DFT amplitude only occurs after nDFT samples = 500/3750 = 0.13s, and for the FFT based method after 512/3750=0.186s. However, at 1.01s, 0.01s after the transition in amplitude, the networks by Duda (2010) (thick gray line), and Lyons and Howard (2021) (yellow dashed line) start trending towards the correct value and would be sufficient to indicate whether a previous adjustment by a control system resulted in an increase, decrease, or no-change to the cost-function. If the FFT method were used (noting that we have attempted to improve the speed

by using 75% overlap), only at 1.094s could an assessment of the efficacy of the controller could be made, which is greater than if a sliding DFT algorithm had been used.

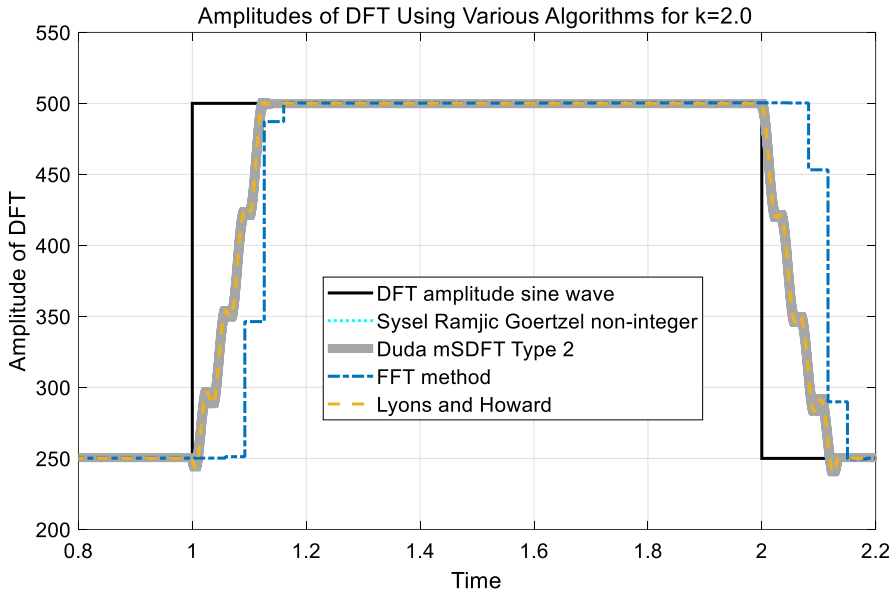

Figure 8: Zoomed version of Figure 7, between time 0.8s to 2.2s, showing comparison of DFT amplitudes.

Next, the excitation and analysis frequencies were changed to 18.75 Hz, that result in a value of $k = 18.75 \times 500/3750 = 2.5$, a non-integer value. The algorithms described above were used to analyse the input signal and the results are shown in Figure 9. The black line indicates the DFT amplitude that the algorithms should calculate. The estimate using the networks by Duda (2010) (gray thick line), and Sysel and Ramjic (2012) (light blue dotted line) diverges after 500/3750 = 0.13s. It can be seen that the algorithm proposed by Lyons and Howard (2021) (yellow dashed line) tracks towards the correct DFT amplitude (black line), which is the desired outcome. The estimate using the FFT method (blue dash-dot line), also tracks towards the correct DFT amplitude (black line), but converges slower than the method by Lyons and Howard (2021).

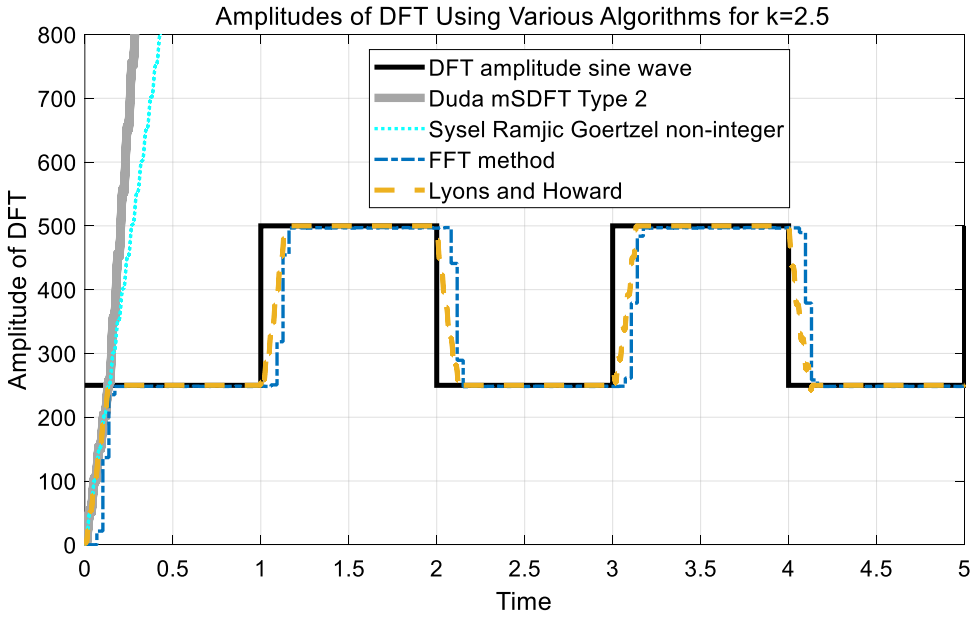

Figure 9: Comparison of DFT amplitudes calculated using various algorithms for k=2.5.

3.2 V8 Diesel Engine Noise

This example involves re-analysing previously published work in Howard and Craig (2014a), Howard and Craig (2014b) of the sound pressure of an 8-cylinder diesel engine that has a capacity of 14-litres, rated power of 350 kW, and was turbo-charged and intercooled. The engine was loaded by a water-brake dynamometer. An adaptive quarter-wavelength tube was attached to a side-branch to the main exhaust pipe. The length of the tube was adjusted until it corresponded to one-quarter of the acoustic wavelength of interest, which caused the amplitude of the sound in the main exhaust duct at the wavelength of interest to be reduced (Craig and Howard, 2012).

When the engine crankshaft was rotating at 1600 rpm, the main cylinder firing frequency was (1600/60)*(8/2) = 106.67 Hz, as each cylinder fires once every two crankshaft revolutions in a four-stroke engine. After digitising the microphone signal at a sample rate of 1600 Hz, the sliding-DFT network shown in Figure 1 was used to monitor the amplitude of the tone at 106.67 Hz as the piston inside the quarter-wavelength tube was fully extended, to simulate an exhaust system without a side-branch and quarter-wavelength tube installed, then gradually retracted. When the position of the AQWT was optimized, it caused the sound pressure level in the exhaust duct to be minimized. For this example, number of DFT points was nDFT = 1024, resulting in a non-integer value of *k* of

$$k = \frac{106.67\ \mathrm{nDFT}}{f_s} = \frac{106.67 \times 1024}{1600} = 68.27 \tag{2}$$

Figure 10 shows the amplitude of the exhaust sound pressure level vs time at 106.67 Hz when the length of the adaptive quarter-wavelength tube was adjusted from the shortest length at time 0 seconds, to simulate an exhaust system without a quarter-wavelength tube installed, and was gradually extended by moving every 1s. The figure shows that at time 130 seconds, the quarter-wavelength tube was tuned and reduced the exhaust sound pressure by roughly 13 dB, which indicates the potential for the AQWT to provide noise reduction of the exhaust tone.

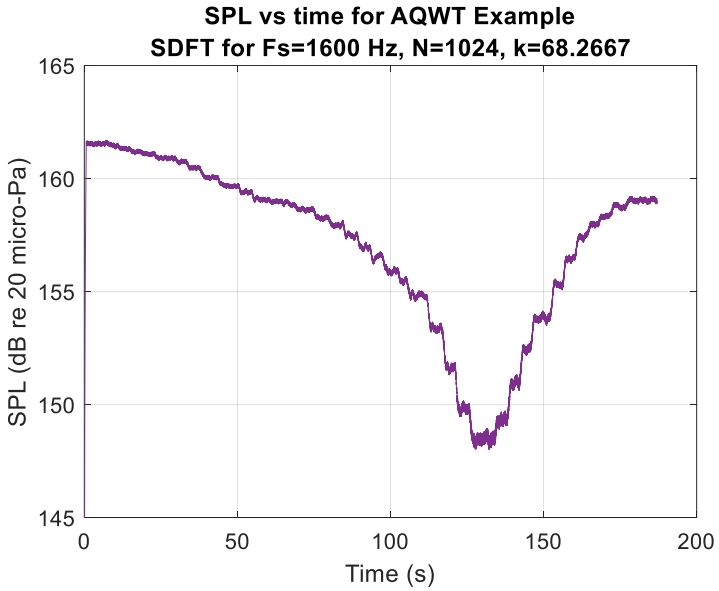

Figure 10: Sound pressure level in the exhaust at 106.67 Hz calculated using the sliding-DFT algorithm in Figure 1, as the length of the AQWT was adjusted every second.

Figure 11 shows corresponding sound pressure level vs frequency when the AQWT was fully extended so the side-branch tube was the shortest, which is effectively the case where the side-branch does not exist, and when the AQWT was optimally tuned to attenuate the tone at the 1x Cylinder Firing Frequency, which was at 106.7 Hz, and shows that the SPL was reduced by 13 dB.

If the proposed sliding-DFT algorithm shown in Figure 1 had been used for this adaptive control system, it would have been able to provide estimates of the amplitude of the tonal noise at 106.7 Hz faster than if an FFT was used for signal monitoring, and would have enabled more rapid control system updates and convergence.

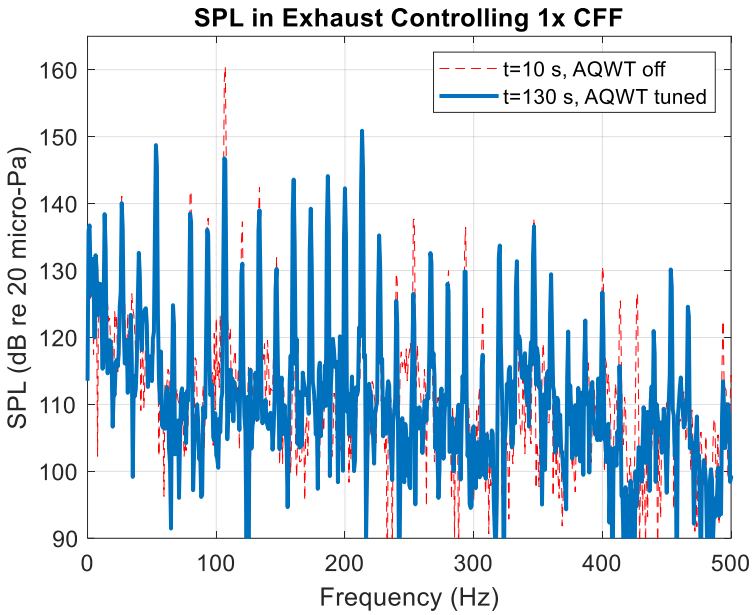

Figure 11: Spectrum of the sound pressure level when the AQWT was fully extended (red dashed line), and when it was optimally tuned (blue thick line).

4 CONCLUSIONS

The results from these simulations of sliding-DFT algorithms show that the proposed algorithm in Figure 1 is able to provide an estimate of the DFT amplitude at non-integer values of k, and has a sliding feature so estimates of the DFT are available at every sample. This is in contrast to other algorithms that have limitations where only non-integer values of k can be used, or they do not have a sliding capability. It was demonstrated that the proposed algorithm was also able to provide an indication of the change in the DFT magnitude when the magnitude of the input signal alters, before nDFT samples had been processed, which enables rapid update of a control system. By using the DFT methods, computational effort is only expended to determine the amplitudes at the frequencies of interest. Reducing the computational effort is vital when implementing real-time control systems as it provides greater time for other non-real-time tasks, or enables greater frequency resolution.

REFERENCES

Chicharo, J. and Kilani, M. (1996). "A Sliding Goertzel Algorithm", Signal Processing, no. 52, pp. 283-297, March. https://doi.org/10.1016/0165-1684(96)00066-7

Craig, R.A. and, Howard, C.Q. (2012). "Development of an Adaptive Quarter-Wave Tube Attached to a Large Diesel Engine", *Proceedings of Acoustics 2012*, Fremantle, Western Australia, pp. 112-118.

Duda, K. (2010). "Accurate, Guaranteed Stable, Sliding Discrete Fourier Transform", IEEE Signal Processing Magazine, vol. 27, no. 6, pp. 124–127, November. https://doi.org/10.1109/MSP.2010.938088

Howard, C.Q. and Craig, R.C. (2014a). "An adaptive quarter-wave tube that uses the sliding-Goertzel algorithm for estimation of phase", Applied Acoustics, 78, pp. 92-97. https://doi.org/10.1016/j.apacoust.2013.12.002

Howard, C.Q. and Craig, R.C. (2014b). "Noise reduction using a quarter wave tube with different orifice geometries", Applied Acoustics, vol. 76, pp. 180-186. https://doi.org/10.1016/j.apacoust.2013.08.006

Howard, C.Q. (2021). "Simulink model of Sliding DFT at exact analysis frequency", MATLAB Central File Exchange, https://www.mathworks.com/matlabcentral/fileexchange/101944-simulink-model-of-sliding-dft-at-exact-analysis-frequency, Retrieved November 12, 2021.

Jacobsen, E. and Lyons, R. (2003). "The sliding DFT", IEEE Signal Processing Magazine, vol. 20, no. 2, pp. 74–80, March. https://doi.org/10.1109/MSP.2003.1184347

Jacobsen, E. and Lyons, R. (2004). "An Update to the Sliding DFT", IEEE Signal Processing Magazine, vol. 21, no. 1, pp. 110–111, January. https://doi.org/10.1109/MSP.2004.1516381

Lyons, R. and Howard, C.Q. (2021). "Improvements to the Sliding Discrete Fourier Transform Algorithm", IEEE Signal Processing Magazine, vol. 38, no. 4, pp. 119-127, July. https://doi.org/10.1109/MSP.2021.3075416

Sysel, P. and Rajmic, P. (2012). "Goertzel algorithm generalized to non-integer multiples of fundamental frequency", EURASIP Journal on Advances in Signal Processing, no. 56, pp. 1-8. https://doi.org/10.1186/1687-6180-2012-56