International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 08, August-2022

VCETDSPDCOM: A Open-Source Python Package to Simulate Signal Processing Algorithms and Design Communication Modulation and Demodulation Schemes

Shivaprasad
Department of EC
VCET, Puttur,
D.K, Karnataka, Puttur

***Abstract*— Python programming language is most widely used language in the recent decades. The language has witnessed a massive number of users due to its easy to comprehend keywords and instructions. There is no Python-based Laboratory for simulating DSP and DCOM experiments in the Electronics and Communication curriculum at VTU Belgaum. The students belonging to this program would be benefited tremendously if the lab experiments related to the course DSP Lab and Communication Lab is designed completely using Python. With these considerations, a package is created using Python that is user-friendly and freely accessible. The students will gain programming knowledge in Python using this package, and they will be better equipped to replicate experiments pertaining to courses DSP-Lab 18ECL57 and Communication -Lab 18ECL67.**

Keywords*— *Vcetdspdcom; DSP; DCOM

I. INTRODUCTION

The open-source package vcetdspdcom, was developed with the intention to benefit students to implement experiments belonging to DSP and D-Com domains using python programming. Python programming is gaining more popularity due to its easiness to code and comprehension. It allows highly complicated problems to be solved using python packages without having to be able to code it from scratch. Essential packages to implement DSP and DCom experiment includes Numpy, Scipy, Matplotlib.

Vcetdspdcom Package consists of Basic/Advanced Signal Processing algorithms and Digital Communication modulation and demodulation techniques. DSP module in the package includes functions to evaluate Linear convolution, Circular convolution, Discrete Fourier transform, Fourier transform of rectangular and sinc pulse, Auto-Correlation, Cross-Correlation, FIR, IIR filter, Discrete cosine transform (DCT), Inverse Discrete cosine transform (IDCT), Discrete wavelet transform (DWT), Inverse Discrete wavelet transform (IDWT), Mel-Frequency cestrum coefficients (MFCC). DCOM module in the package includes functions to evaluate important Digital modulation and demodulation techniques like Amplitude shift keying (ASK), Frequency shift keying (FSK), Phase shift keying (PSK), Differential Phase shift keying (DPSK), Quadrature Phase shift keying (QPSK), M-ary Quadrature Amplitude Modulation (Eg: 16-QAM) and Time Division Multiplexing (TDM).

II. GETTING STARTED WITH PACKAGE

A. Installing Package

Vcetdspdcom Package could be installed on your windows OS by typing the command,**pip install vcetdspdcom**, in windows command prompt(ensure that python interpreter is already installed).

B. Importing package in Interactive mode

Once the package is installed in your system, we can import it by typing the command import vcetdspdcom in the python command prompt. If everything is fine, the cursor position in the command prompt goes to the next line. This implies the package is installed as well as imported successfully.

C. Testing the modules included in the package using Interactive mode

The package includes two modules dsp and dcom respectively. Each module includes several functions defined in it.

(i) To use the function linear_conv belonging to the module dsp under the package vcetdspdcom, we use the command

```
>>>import vcetdspdcom.dsp as dsp
>>>dsp.linear_conv([1,2],[1,1]))
```

In the above code snippet, x(n)={1,2} and h(n)={1,1} with length M=2 and N=2. The linear convolution of x(n) with h(n) gives rise to y(n)={1,3,2}

(ii) To use the function ask_modulation belonging to the module dcom under the package vcetdspdcom, we use the command

```
>>>import vcetdspdcom.dcom as dcom
>>>fs=100
>>>fc=5
>>>tb=0.1
>>>sim_t=1
>>>[y,x,c,t]=dcom.ask_modulation(fs,fc,tb,sim_t)
```

In the above code snippet, sampling frequency=100Hz, carrier signal frequency=5Hz, bit duration=0.1 seconds and simulation time=1 seconds. The returned variables y, x, c and t represents ASK modulated signal, input message

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 08, August-2022

signal, carrier signal and time. These variables could be used to plot the graph for ASK analysis

III. DSP EXPERIMENTS SIMULATION USING VCETDSPDCOM

In this section, usage of some of important algorithms pertaining to signal processing domain are discussed.

A. *Sampling theorem*

Sampling theorem program function included in the package takes inputs as resolution of analog signal (0.001 seconds), fundamental frequency of signal (5Hz) and sampling frequency (100 Hz) and returns output parameters including time interval of analog signal(t), input signal(x), discrete sequence(xn), time interval of discrete sample(n) and reconstructed signal(xr). Package uses interpolation filter to reconstruct original signal from its samples. The following code snippet is used to simulate sampling theorem [1],[4],[7],[8].

```
>>>import vcetdspdcom.dsp as dsp
>>>[t,x,n,xn,xr]=dsp.sampling_theorem()
```

Fig.1 below depicts the results obtained using sampling_theorem function which includes original signal, sampled signal and recovered signal from samples using interpolation filter.

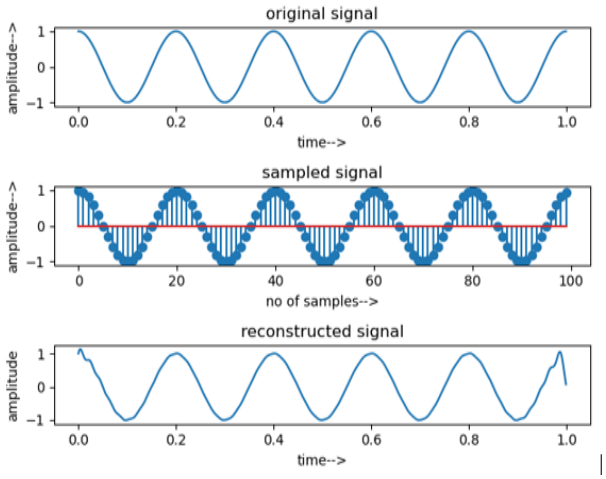

Fig. 1. Sampling theorem analysis

B. *Fourier transform of rectangular pulse*

FT of rectangular function available in the dsp module takes input arguments as duration of rectangular pulse(1 seconds), amplitude of the pulse(1 volt), sampling frequency(1000 Hz) and returns time domain signal (x), time domain range(t), Frequency response of x(X), frequency domain range(f). The following code snippet is used to simulate FT of rectangular pulse[1],[3],[7],[8].

```
>>>import vcetdspdcom.dsp as dsp
>>>du=input('enter the duration of rectangular pulse')
>>>a=input('enter the amplitude')
>>>fs=input('enter the sampling frequency')
>>>[x,t,X,f]=dsp.rect2sinc(du,a,fs).
```

Fig.2 below shows the results obtained using rect2sinc function which includes rectangular pulse with lasts for one seconds and its fourier transform

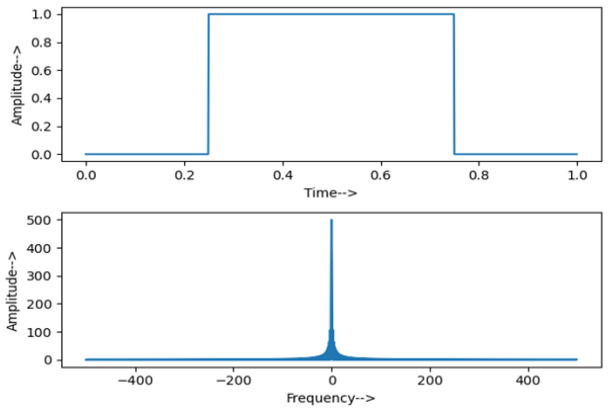

Fig.2.Time and Frequency domain analysis of rectangular pulse

C. *Design of Low Pass FIR filter using windows*

Low Pass FIR filter function available in the dsp module takes input such as order N(51), cut-off frequency in radians wc(π/4), window function win(hamm), frequency resolution f_res(0.001 seconds) and returns output arguments which includes impulse response of LP filter(h), frequency resolution(t), Magnitude of frequency response(H). The following code snippet is used to simulate Low Pass FIR filter [1],[2],[4],[7],[8]..

```
>>>import vcetdspdcom.dsp as dsp
>>>N=int(input('enter order'))
>>>wc=float(input('enter cut off freq'))
>>>win=input('enter window')
>>>f_res=float(input('enter frequency resolution'))
>>>[h,t,H]=dsp.fir_lpf(N,wc,win,f_res)
```

Fig.3. below shows the impulse response and frequency response of Low pass FIR filter obtained for certain specifications using hamming window

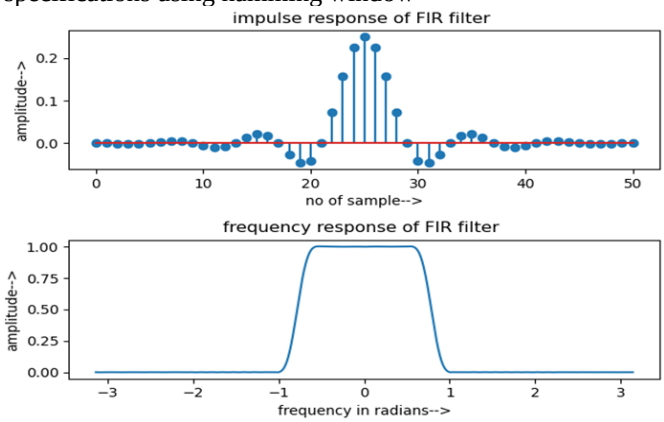

Fig. 3. Low FIR filter impulse and frequency respons

D. *MFCC coefficients of speech signal*

Mel frequency cestrum coefficients of speech signal is most widely adapted feature extraction technique used to acquire important features from speech signal. The input arguments passed as an argument to the function mfcc includes speech signal(mono audio signal), sampling frequency(fs),

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 08, August-2022

frame length(512), frame step(256) and number of Mel-filter banks(18). The output arguments returned by the function includes 18 MFCC coefficients. The following code snippet explains the same [1],[3],[7],[8].

```
<<<import vcetdspdcom.dsp as dsp
<<<from scipy.io import wavfile
<<<fs,a=wavfile.read('for1.wav')
<<<a=a[:,0]
<<<frame_length=512
<<<frame_step=256
<<<filter_bank_size=18
    <<<mfc=mfcc_coeff(a,fs,frame_length,frame_step,filter_b
        ank_size)
```

IV. DCOM EXPERIMENTS SIMULATION USING VCETDSPDCOM

In this section, usage of some of important algorithms pertaining to communication domain are discussed.

A. ASK Modulation and Demodulation

Amplitude-shift keying modulation function available in dcom module requires input parameters which includes sampling frequency (1000 Hz), carrier frequency (10 Hz), bit duration (0.1) and simulation time (1 seconds). The output parameters returned by the modulation function are ASK-modulated signal, input message, carrier signal and resolution of input and output signals to plot.

Amplitude-shift keying demodulation function in dcom module takes input parameter as ASK-modulated signal, sampling frequency, carrier frequency and bit duration and returns output arguments as demodulated signal and demodulated data bits. Code snippet for the above modulation and demodulation process is given below[1],[5],[6],[7].

```
<<<import vcetdspdcom.dcom as dcom
<<<fs=1000#sampling frequency
<<<fc1=10#carrier frequency-1
<<<tb=0.1#bit duration
<<<sim_t=1#simulation time
<<<[y_output,message,c_carrier1,t_time]=dcom.ask_modulati
on(fs,fc1,tb,sim_t)
<<<[rec_signal,rec_bits]=dcom.ask_demodulation(y_output,fs
,c_carrier1,tb)
```

Fig. 4. displays the result obtained using the ask_modulation and ask_demodulation function which includes message signal, carrier signal, ASK- modulated and demodulated signal.

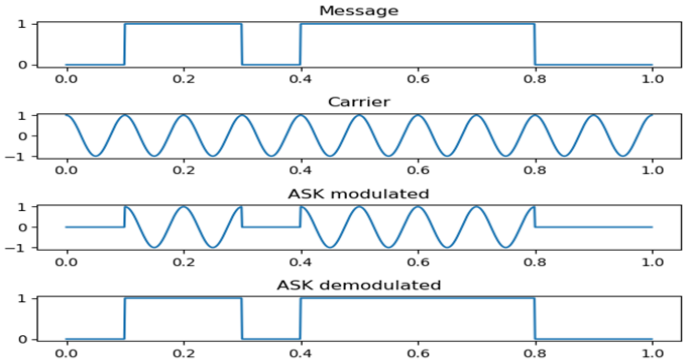

Fig.4. ASK modulation and demodulation

B. DPSK Modulation and Demodulation

Differential Phase shift keying modulation function available in dcom module requires input parameters which includes sampling frequency (1000 Hz), carrier frequency (5 Hz), bit duration (0.1 seconds) simulation time (1 seconds) and initial start bit (1). The output parameters returned by the modulation function are DPSK-modulated signal, input message, encoded message, carrier frequency and resolution of input and output signals to plot

Differential Phase shift keying demodulation function in dcom module takes input parameter as DPSK-modulated signal, sampling frequency and bit duration and returns output arguments as demodulated signal and demodulated data bits. Code snippet for the above modulation and demodulation process is given below[1],[5],[6],[7]..

```
<<<import vcetdspdcom.dcom as dcom
<<<fs=1000#sampling frequency
<<<fc1=5#carrier frequency-1
<<<tb=0.1#bit duration
<<<sim_t=1#simulation time
<<<init_bit=1
<<<[y_output,message,enc_message,c_carrier1,t_time]=dcom
.dpsk_modulation(fs,fc1,tb,sim_t,init_bit)
<<<[rec_signal,rec_bits]=dcom.dpsk_demodulation(y_output,
fs,tb)
```

Fig. 4. displays the result obtained using the dpsk_modulation and dpsk_demodulation function which includes message signal, encode message signal, carrier signal, DPSK-modulated and demodulated signal

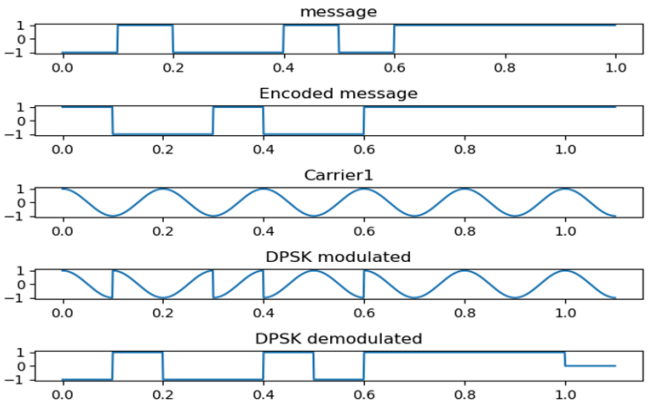

Fig.5. DPSK modulation and demodulation process

C. QPSK Modulation and Demodulation.

Quadrature Phase shift keying modulation function available in dcom module requires input parameters which includes sampling frequency (1000 Hz), carrier frequency (5 Hz), bit duration (0.1 seconds) and simulation time (1 seconds). The output parameters returned by the modulation function are QPSK-modulated signal, even signal, odd signal, carrier frequency and resolution of input and output signals to plot.

Quadrature Phase shift keying demodulation function in dcom module takes input parameter as QPSK-modulated signal, sampling frequency, carrier frequency and bit duration and returns output arguments as demodulated signal and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 08, August-2022

demodulated data bits. Code snippet for the above modulation and demodulation process is given below[1],[5],[6],[7].

```
<<<import vcetdspdcom.dcom as dcom
<<<fs=1000#sampling frequency
<<<fc1=5#carrier frequency-1
<<<tb=0.1#bit duration
<<<sim_t=1#simulation time
<<<[y_output,msg,ev_signal,od_signal,c_carrier1,t_time]=dcom.qpsk_modulation(fs,fc1,tb,sim_t)
<<<[dmsg,rec_bits]=dcom.qpsk_demodulation(y_output,fs,fc1,tb)
```

Fig. 6. displays the result obtained using the qpsk_modulation and qpsk_demodulation function which includes message signal, odd signal, even signal, carrier signal, QPSK modulated and demodulated signal.

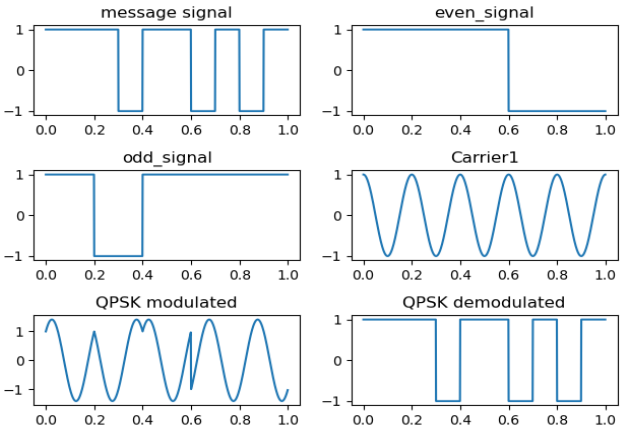

Fig.6. QPSK modulation and demodulation

D. TDM Modulation and Demodulation.

Time division multiplexing function available in dcom module requires input parameters which includes sampling frequency (1000 Hz), sinusoidal message signal-1(2 Hz), sinusoidal message signal-2(6 Hz), control signal bit duration (0.01 seconds) and simulation time (1 seconds). The output parameters returned by the TDM modulation function are TDM multiplexed signal, message-1, message-2, control signal and resolution of input and output signals to plot.

TDM demodulation function in dcom module takes input parameter as TDM-multiplexed signal, sampling frequency and control signal bit duration and returns output arguments as demodulated message-1 and demodulated message-2. Code snippet for the above modulation and demodulation process is given below[1],[5],[6],[7].

```
<<<import vcetdspdcom.dcom as dcom
<<<sim_t=0.5
<<<tcb=0.01
<<<fs=1000
<<<fc1=2
<<<fc2=6
<<<fs=1000
<<[y_output,msg1,msg2,control_signal,t_time]=dcom.tdm_modulation(fs,fc1,fc2,tcb,sim_t)
<<<[d1,d2]=dcom.tdm_demodulation(y_output,fs,tcb)
```

Fig.7. displays the result obtained using the TDM multiplexing and demultiplexing function which includes message signal-1, message signal-2, control signal, TDM multiplexed and demultiplexed signal.

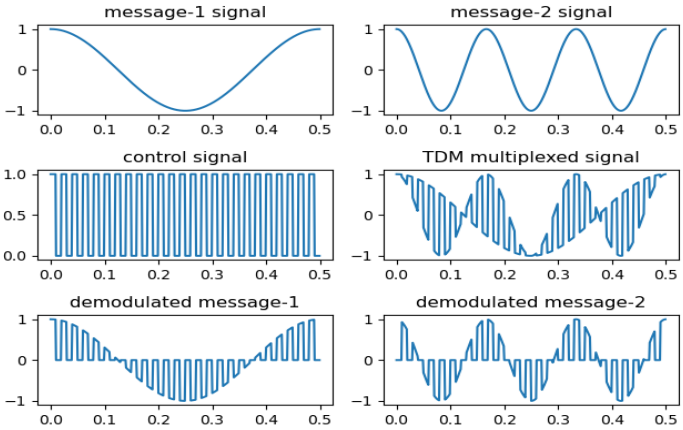

Fig.7. TDM multiplexing and demultiplexing waveforms

V CONCLUSION

Open-source package developed using python programming provides various essential algorithms implemented in the area of signal processing and communication. Students will find these packages helpful as they can design basic algorithms in DSP and Dcom domain using fewer lines of codes. Students could use these features to conduct the experiments and record the results. Python programming basics are not essential to design the above said algorithms.

REFERENCES

[1] Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). DOI: 10.1038/s41586-020-2649-2.

[2] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E.A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. (2020) SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17(3), 261-272.

[3] Z. Wanli and L. Guoxin, "The research of feature extraction based on MFCC for speaker recognition," Proceedings of 2013 3rd International Conference on Computer Science and Network Technology, 2013, pp. 1074-1077, doi: 10.1109/ICCSNT.2013.6967289.

[4] Vinay K. Ingle and John G Proakis. 2011. Digital Signal Processing Using MATLAB (3rd.ed.).CL-Engineering

[5] McKinney, W., & others. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56).

[6] Haykin, Simon. Communication systems. John Wiley & Sons, 2008.

[7] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90–95.

[8] J.G. Proakis, D.G. Manolakis, Digital Signal Processing Principles, Algorithms, and Applications, 3rd edn. (Prentice-Hall, Englewood Cliffs,1996)