Summer Fellowship Report

On

Scilab Case Study and Xcos TBC

Submitted by

Kshitiz

Electronics and Communication Department

IIIT Senapati,Manipur

Under the guidance of

Prof.Kannan M. Moudgalya

Chemical Engineering Department

IIT Bombay

Mentors

Ms. Rashmi Patankar

Ms. Vineeta Ghavri

Mr. Rupak Rokade

June 10, 2020

3

Acknowledgment

The fellowship opportunity we had with the FOSSEE Team, IIT BOMBAY, was a great chance for learning and professional development. Therefore, we consider ourselves as very lucky individuals as we were provided with an opportunity to be a part of it. We are also grateful for having a chance to meet so many wonderful people and professionals across the country who led us through this internship period.

We are using this opportunity to express our deepest gratitude and special thanks to Prof. Kannan M. Moudgalya, head of FOSSEE team, IIT Bombay, for giving us an opportunity to be a part of this project.

It is our radiant sentiment to place on record our best regards, deepest sense of gratitude to our mentors, Ms.Rashmi Patankar, Ms.Vineeta Ghavri and Mr.Rupak Rokade for their continuous support which were extremely valuable for our study both theoretically and practically and helping us to learn a lot many things.

We perceive this opportunity as a big milestone in our career development. We will strive to use gained skills and knowledge in the best possible way, and we will continue to work on their improvement, in order to attain desired career objectives. Hope to continue cooperation with all of you in the future.

Contents

Chapter 1

Introduction

1.1 Overview

The fellowship task was sub divided into two halves with first being the Case Study 2 which was to be done during the given duration and providing proof of the concept for validation part, which can actually vary in many proven and accepted ways namely either by **submitting the reference paper or trying the code over a lot of examples or both**.

The second half of the fellowship was to make Xcos TBC (Text Book Companion) in which we were asked to code 5 books in Xcos to make the documentation more profound.It may also help the user to get more familiar with Xcos blocks as well as it will be useful in further modeling of different system using it.It has been my earnest effort to show the potential of the Xcos in various ways and also how to utilize its toolboxes to the fullest.

Since in FOSSEE fellowship the stress was laid upon that up-to what extent your solution to a problem is acceptable and feasible. And also the code should be maintained in such a way that any reader can easily familiarize to it and in order to do so it was necessary to make documentation as lucid as possible.

Chapter 2

Technical Specification

2.1 Scilab

Scilab is free and open source software for numerical computation providing a powerful computing environment for engineering and scientific applications.
Scilab includes hundreds of mathematical functions. It has a high level programming language allowing access to advanced data structures, 2-D and 3-D graphical functions.

A large number of functionalities is included in Scilab:

- **Maths & Simulation**
  For usual engineering and science applications including mathematical operations and data analysis.
- **2-D & 3-D Visualization**
  Graphics functions to visualize, annotate and export data and many ways to create and customize various types of plots and charts.
- **Optimization**
  Algorithms to solve constrained and unconstrained continuous and discrete optimization problems.
- **Statistics**
  Tools to perform data analysis and modeling
- **Control Systems**
  Standard algorithms and tools for control system study
- **Signal Processing**
  Visualize, analyze and filter signals in time and frequency domains.
- **Application Development**
  Increase Scilab native functionalities and manage data exchanges with external tools.
- **Xcos - Dynamic systems modeling**
  Modeling mechanical systems, hydraulic circuits, control systems...

2.1.1 IPCV Toolbox

Install command - atomsInstall(”IPCV”)

IPCV – Scilab Image Processing Computer Vision, a module of Image Processing and Computer Vision Toolbox for Scilab 6.0.

2.1.2 Scilab on Cloud

Scilab on Cloud facilitates execution of the codes for particular example(s) online. The results can then be verified with the solved example(s) from the textbook.Follow the link to view Scilab-on-Cloud

2.2 Xcos

Xcos is a graphical editor to design hybrid dynamical systems models. Models can be designed, loaded, saved, compiled and simulated. Ergonomic and efficient solution for industrial and academics needs, Xcos provides functionalities for modeling of mechanical systems (automotive, aeronautics...), hydraulic circuits (dam, pipe modeling...), control systems, etc.

Xcos is freely available and distributed with Scilab.

- Standard Palettes and Blocks
- Model building and edition

- Model Customization and Modelica blocks creation Simulation

2.2.1 CPGE Toolbox

Install command - atomsInstall(”CPGE”)

This module provides Xcos blocks used in the study of systems for the teaching of IBS in preparatory classes for grandes écoles. It allows to : - model slave systems, - do time simulation (multi-parameter time analysis), - do frequency simulation (Bode, Black and Nyquist diagrams), - carry out systems studies (MAXPID robot)

2.2.2 Xcos on Cloud

Xcos on cloud provides the browser-only version to try Xcos functionalities without installing native Scilab software.Follow on the link to view the Xcos-on-Cloud

Chapter 3

Scilab Case Study 1(Audio Processing)

3.1 Noise Reduction Algorithm

3.1.1 Abstract

Many of the times when we record the actual voice sample using microphone there is unwanted noise which is introduced due to the surrounding which makes the original signal distorted and make it unclear to understand. This happens with many high end devices such as(microphone of smart phones, laptops etc) .This can be reduced using audio processing techniques mainly by filtering the unwanted segments from the audio sample to make it much clear. Along with that various techniques can be used to further enhance the quality as well as adding Audio ejects to the sample like Echoing,Digital Reverberation.

3.1.2 Problem Statement

A voice sample of Navy officer's Speech[and other similar noisy signal] is taken in which is recorded on microphone where noise can be easily identified.The voice sample is of length 39 seconds. As he tries to deliver the message to its recording is full of noticeable noise which is introduced due to surroundings. Task is to suppress the noise as much as possible without attenuating the information of the Navy Officer [and various other] by using various techniques and improving the pitch if possible for decent audible experience. And also to show the time and frequency domain plot for each step of suppression along with the original audio plot.

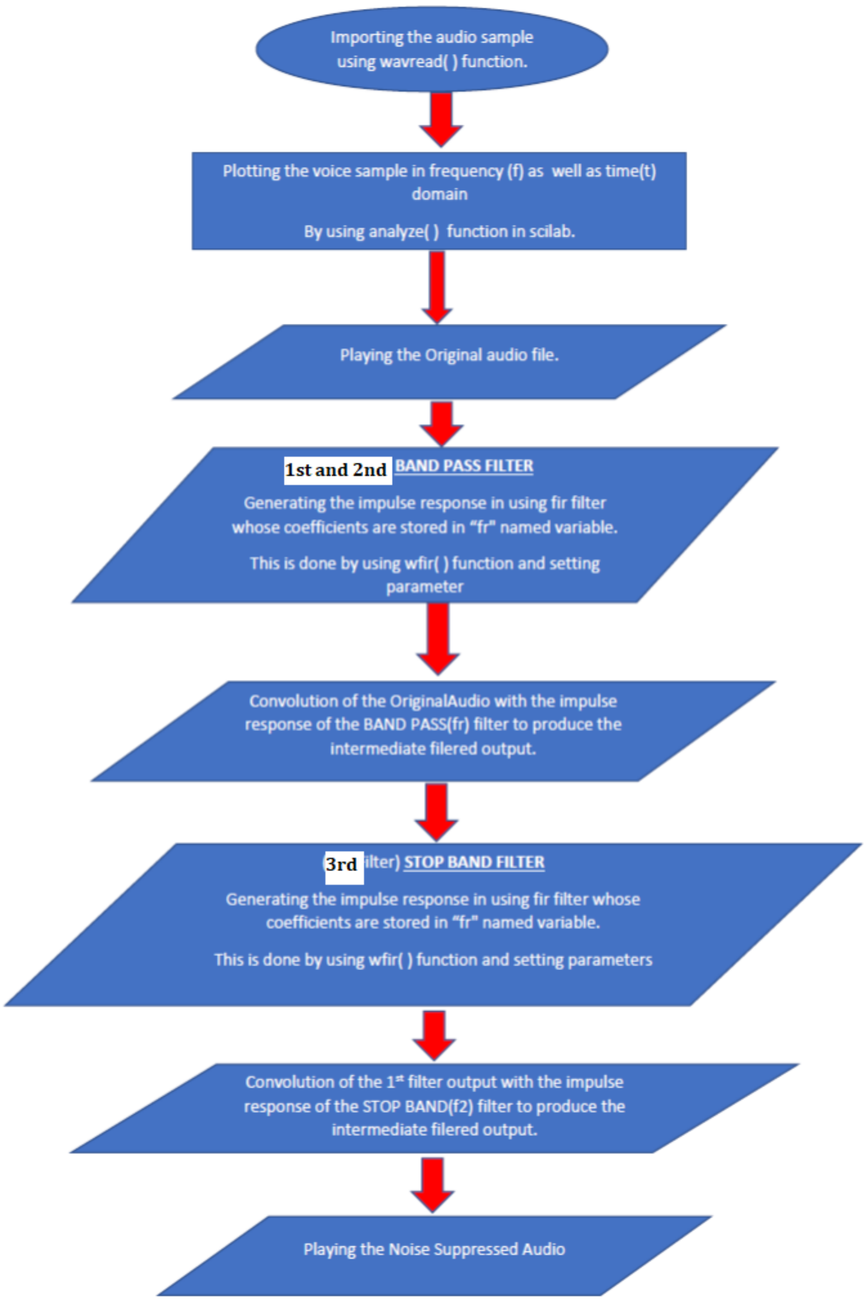
3.1.3 Objective

It is mainly designed for the emergency situation such as transmission on the aircraft carrier where fighter planes take off .Recording of the samples are from different environment and equipment and due to noisy content output may have attenuation from 20% to 40% due to prevailing circumstances and recording devices. The main motive is to clearly identify the message in the audio sample.

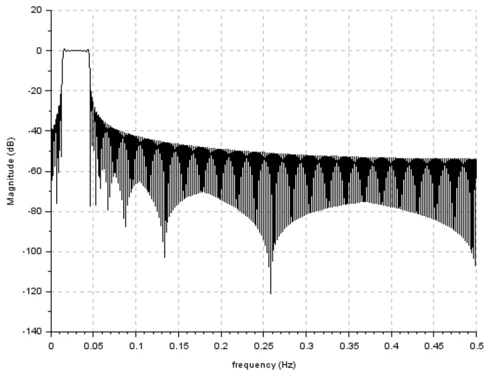
3.1.4 Approach

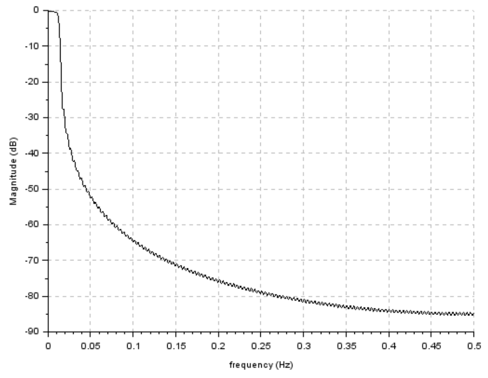
The model is based on the extraction of sound lying in the range of human vocal chord frequency range with some deviation due to noise. Which require suppression of the noise using various filters and also enhancing the audio quality if processing.

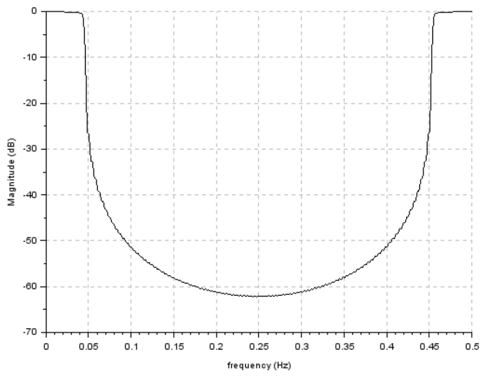
3.1.5 Flowchart


Importing the audio sample using wavread( ) function.
Plotting the voice sample in frequency (f) as well as time(t) domain
By using analyze( ) function in scilab.
Playing the Original audio file.
1st and 2nd BAND PASS FILTER
Generating the impulse response in using fir filter whose coefficients are stored in "fr" named variable.
This is done by using wfir( ) function and setting parameter
Convolution of the OriginalAudio with the impulse response of the BAND PASS(fr) filter to produce the intermediate filered output.
3rd ilter) STOP BAND FILTER
Generating the impulse response in using fir filter whose coefficients are stored in "fr" named variable.
This is done by using wfir( ) function and setting parameters
Convolution of the 1st filter output with the impulse response of the STOP BAND(f2) filter to produce the intermediate filered output.
Playing the Noise Suppressed Audio

Filters Response of different filters are shown below



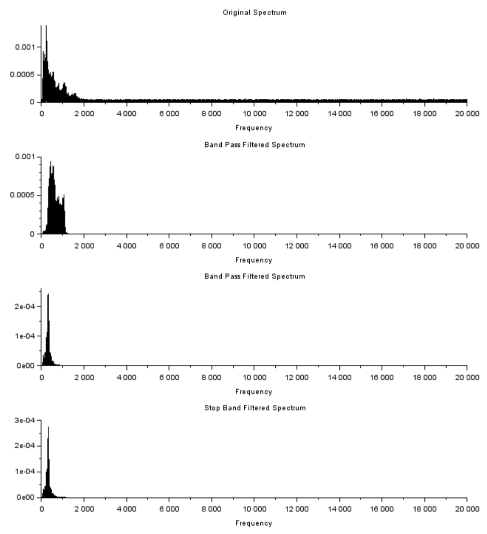


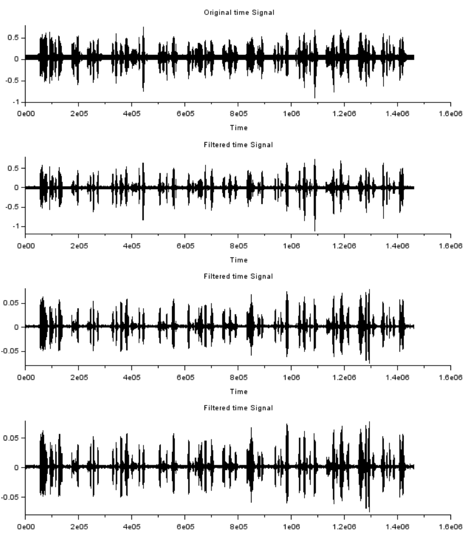


Filter at top left is the Band Pass filter (600 Hz to 2000 Hz) normalized to [0-1] ,in top right is the Band Pass filter (0Hz to 600Hz) normalized to [0-1] and at bottom is the Stop Band filter (2000 Hz to 20000 Hz) with hamming window normalized to [0-1]

3.1.6 Output

Output consist of several audio file which is played by just executing the **Case_study1.sce** script. It is similar to an auto-run file. The visual analysis can be done by just looking at the plot that pops out representing the time and frequency domain plot of the actual audio signal.





Chapter 4

Scilab Case Study 2(Image Processing)

4.1 Cloud Removal Algorithm from Satellite Imagery

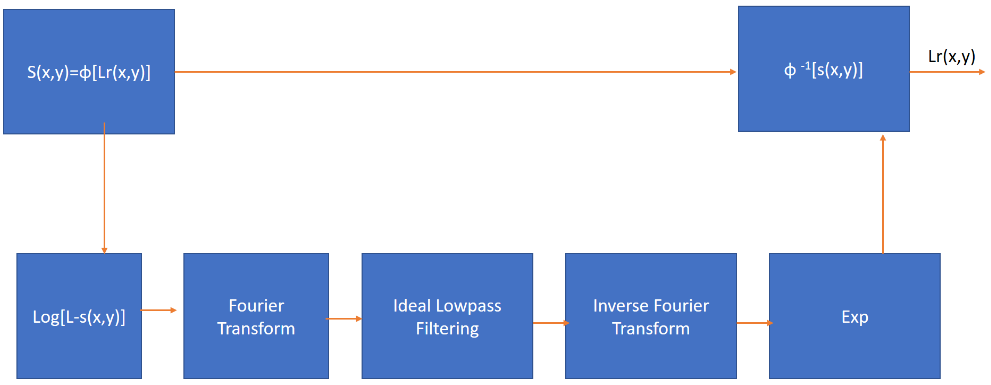
4.1.1 Abstract

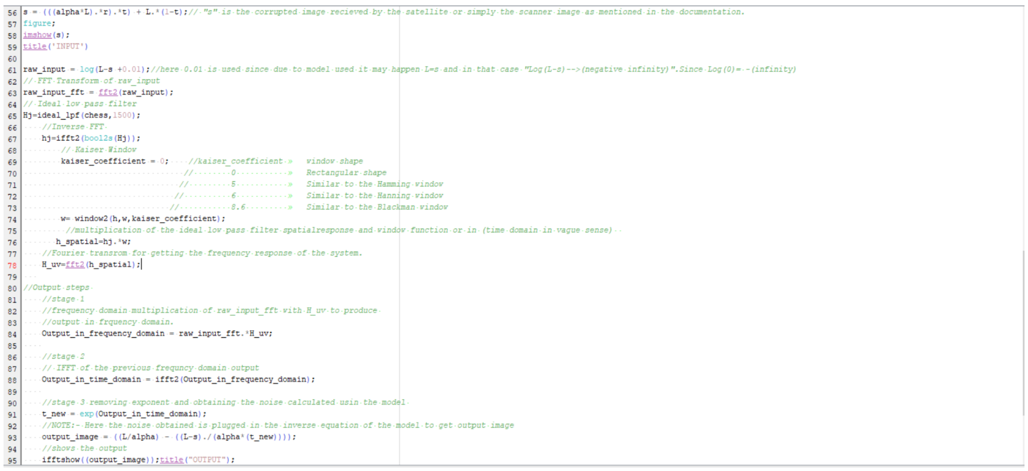
The project aims to simulate the cloud cover noise from thin to thick and removal of it for revealing the objects beneath the clouds.The average cloud coverage for the entire world is about 40 percent .If the cloud cover is too thick, recovery of the underlying signal is, of course, not possible.

4.1.2 Problem Statement

Cloud noise is not strictly additive or multiplicative but rather the combination of both. Then by using the optimum filtering techniques removal of the clouds can be done.In this simulation the data which gets lost due to noise is to be estimated based on the natural parameters like cloud thickness,sun illumination and sun- light attenuation.

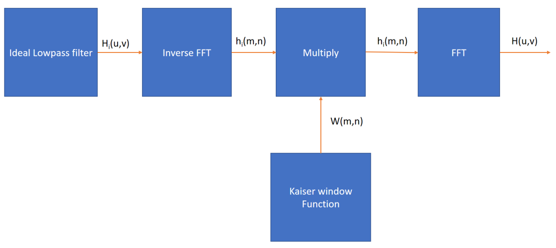
4.1.3 Schematic Workflow Diagram | Codes Snippet

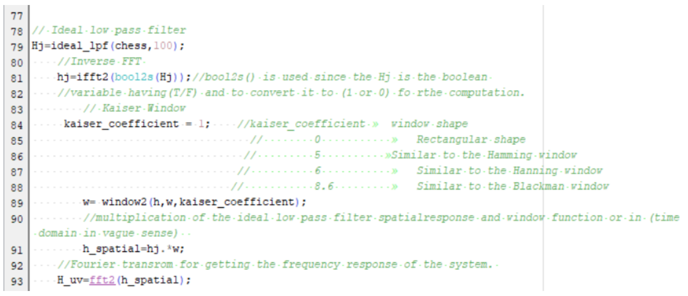




```
s = (((alpha*L).*r).*t) + L.*(1-t);// "s" is the corrupted image recieved by the satellite or simply the scanner image as mentioned in the documentation.
figure;
imshow(s);
title('INPUT')

raw_input = log(L-s +0.01);//here 0.01 is used since due to model used it may happen L=s and in that case "Log(L-s)-->(negative infinity)".Since Log(0)= -(infinity)
// FFT Transform of raw_input
raw_input_fft = fft2(raw_input);
// Ideal low pass filter
Hj=ideal_lpf(chess,1500);
    //Inverse FFT
    hj=ifft2(bool2s(Hj));
        // Kaiser Window
        kaiser_coefficient = 0;    //kaiser_coefficient »   window shape
                                   //        0           »   Rectangular shape
                                  //         5           »   Similar to the Hamming window
                                 //          6           »   Similar to the Hanning window
                                //           8.6         »   Similar to the Blackman window
        w= window2(h,w,kaiser_coefficient);
         //multiplication of the ideal low pass filter spatialresponse and window function or in (time domain in vague sense)
       h_spatial=hj.*w;
    //Fourier transrom for getting the frequency response of the system.
    H_uv=fft2(h_spatial);

//Output steps
    //stage 1
    //frequency domain multiplication of raw_input_fft with H_uv to produce
    //output in frquency domain.
    Output_in_frequency_domain = raw_input_fft.*H_uv;

    //stage 2
    // IFFT of the previous frequncy domain output
    Output_in_time_domain = ifft2(Output_in_frequency_domain);

    //stage 3 removing exponent and obtaining the noise calculated usin the model
    t_new = exp(Output_in_time_domain);
    //NOTE:- Here the noise obtained is plugged in the inverse equation of the model to get output image
    output_image = ((L/alpha) - ((L-s)./(alpha*(t_new))));
    //shows the output
    ifftshow((output_image));title("OUTPUT");
```

Schematics of Algorithm and its realized code snippet



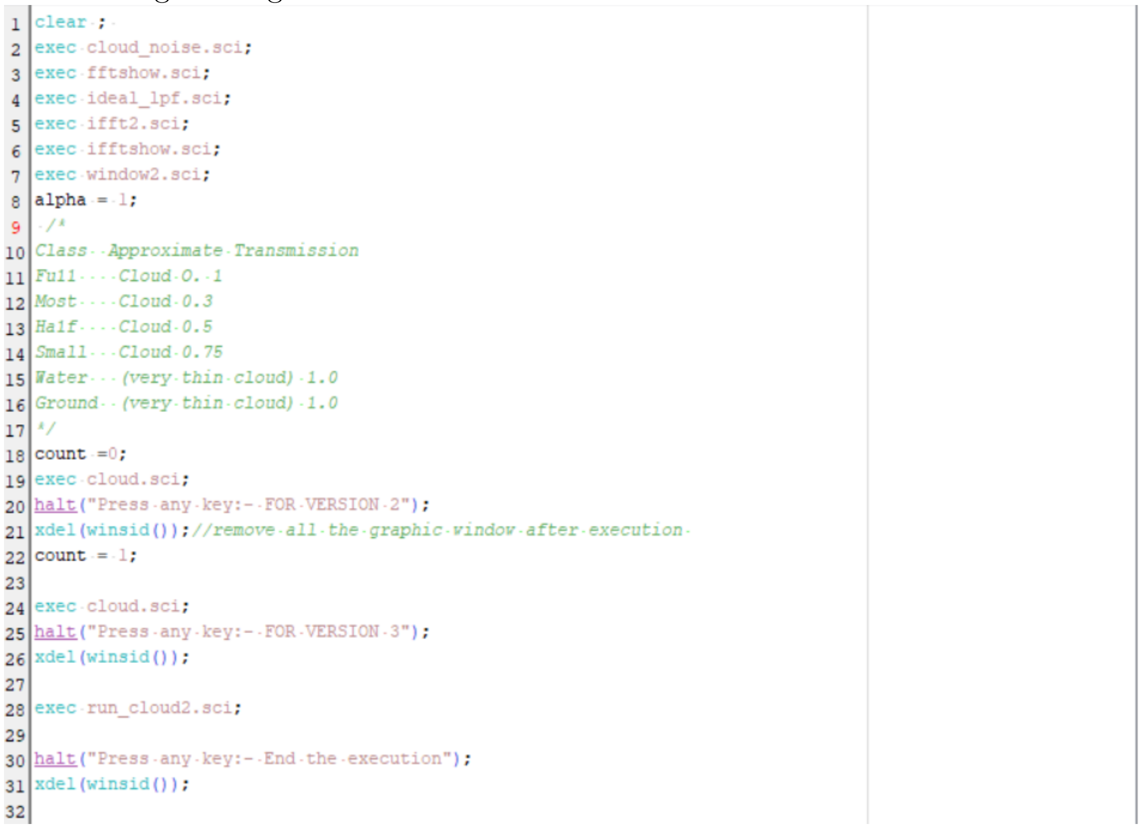


```

// Ideal low pass filter
Hj=ideal_lpf(chess,100);
    //Inverse FFT
    hj=ifft2(bool2s(Hj));//bool2s() is used since the Hj is the boolean
    //variable having(T/F) and to convert it to (1 or 0) fo rthe computation.
        // Kaiser Window
     kaiser_coefficient = 1;    //kaiser_coefficient »  window shape
                                  //        0          »   Rectangular shape
                                 //         5          »Similar to the Hamming window
                                //          6          »   Similar to the Hanning window
                               //           8.6        »   Similar to the Blackman window
        w= window2(h,w,kaiser_coefficient);
        //multiplication of the ideal low pass filter spatialresponse and window function or in (time
 domain in vague sense)
        h_spatial=hj.*w;
    //Fourier transrom for getting the frequency response of the system.
    H_uv=fft2(h_spatial);
```

Filter design schematics and its realized code snippet

4.1.4 Autorun file(Case_Study2.sce)

This file is used to automate the task and used to reduce the clutter of making changes in various other dependency file parameters, making it a one stop portal.All the versions can be executed by just executing it.Cloud thickness parameter(alpha) can be changed using this file.



```
clear ;
exec cloud_noise.sci;
exec fftshow.sci;
exec ideal_lpf.sci;
exec ifft2.sci;
exec ifftshow.sci;
exec window2.sci;
alpha = 1;
 /*
Class  Approximate Transmission
Full    Cloud 0. 1
Most    Cloud 0.3
Half    Cloud 0.5
Small   Cloud 0.75
Water   (very thin cloud) 1.0
Ground  (very thin cloud) 1.0
*/
count =0;
exec cloud.sci;
halt("Press any key:- FOR VERSION 2");
xdel(winsid());//remove all the graphic window after execution
count = 1;

exec cloud.sci;
halt("Press any key:- FOR VERSION 3");
xdel(winsid());

exec run_cloud2.sci;

halt("Press any key:- End the execution");
xdel(winsid());
```

4.1.5 Dependencies

```
//This function generates the noise similar to the noise generated
    by the cloud.
function [out] = cloud_noise(im,D,n)
//im = image passed in as parameter
//D = cutoff frequency
//n = order of the filter
h = size(im,1);// height of the image
w = size(im,2);// width of the image
// Here we are making the filter according to the width of the
    fiter
[x ,y] = meshgrid(-floor(w/2):floor((w-1)/2),-floor(h/2):floor((h
    -1)/2));
out  = 1./(1+(D./(x.^2+y.^2).^0.5).*(2*n));//butter worth filter
    equation.
out = 1-out;
figure;
imshow(out);
```

```

end
```

Listing 4.1: Cloud Like Noise Generation(cloud_noise.sci)

```
//This function is used to show the  frequency domain response of
   the picture
function [ ] = fftshow(f)
//   Detailed explanation goes here
f1 = log(1+abs(f));
fm = max(f1(:));
figure,imshow(im2uint8(f1/fm));
end
```

Listing 4.2: Produces Frequency Domain Response of the Images (fftshow.sci)

```
//This function is used to make the 2D window for the filter.
function w=window2(N,M,alpha)

wc=window('kr',N,alpha);
wr=window('kr',M,alpha);
[maskr,maskc]=meshgrid(wr,wc);

//maskc=repmat(wc,1,M); Old version
//maskr=repmat(wr',N,1);


w=maskr.*maskc;

end
```

Listing 4.3: Generates the 2D Window for the filters (window2.sci)

```
//This function is used for making the Ideal Low Pass Flter.
function [out] = ideal_lpf(im,D)
    //im = image passed in as parameter to the function
    //D = cutoff of the LPF filter.
    h = size(im,1);// height of the image
    w = size(im,2);// width of the image
    // Here we are making the filter according to the width and
   height of the fiter
    [x ,y] = meshgrid(-floor(w/2):floor((w-1)/2),-floor(h/2):floor
   ((h-1)/2));
    z = sqrt(x.^2+y.^2);
    out=z<=D;//D is the  cutoff frequency;
    //figure();
    //imshow(out);

endfunction
```

Listing 4.4: Ideal Lowpass Filter for Images(ideal_lpf.sci)

```
//This function is used to show the image in appropriate form after
    obtaining the the inverse fft from homorphic filtering
   technique
function [new_image] = ifftshow(f)
//f = time domain input which is complex in nature
```

```
f1 = abs(f);//magnitude of the matrix is cazlculated
fm = max(f1(:));//max value is found out from the matrix
new_image=im2uint8(f1/fm);//converted into uint8 data type for
    image to be displayed.
figure,imshow(new_image);
end
```

Listing 4.5: Inverse Fourier Transform of Images(ifftshow.sci)

```
//This function is used to split thr Red,Green,Blue channel of the
    RGB image for further calculations.
function [redchannel,bluechannel,greenchannel] = RGB_splitter(im)
rgbImage = im;
// Extract color channels.
redchannel = rgbImage(:,:,1); // Red channel
greenchannel = rgbImage(:,:,2); // Green channel
bluechannel = rgbImage(:,:,3); // Blue channel
end
```

Listing 4.6: Used to split the Red Green Blue Channel of the RGB images (RGB_recombiner.sci)

```
//This function is used to recombine the red,green,blue channels
    together to form the RGB image.
function [recombinedRGBImage] = RGB_recombiner(redchannel,
    bluechannel,greenchannel,rgbImage)
// Create an all black channel.
allBlack = im2uint8(zeros(size(rgbImage, 1), size(rgbImage, 2)));
// Create color versions of the individual color channels.
just_red = cat(3, im2uint8(redchannel), allBlack, allBlack);
just_green = cat(3, allBlack, im2uint8(greenchannel), allBlack);
just_blue = cat(3, allBlack, allBlack, im2uint8(bluechannel));

// Recombine the individual color channels to create the original
    RGB image again.
recombinedRGBImage = cat(3, redchannel, greenchannel, bluechannel);
end
```

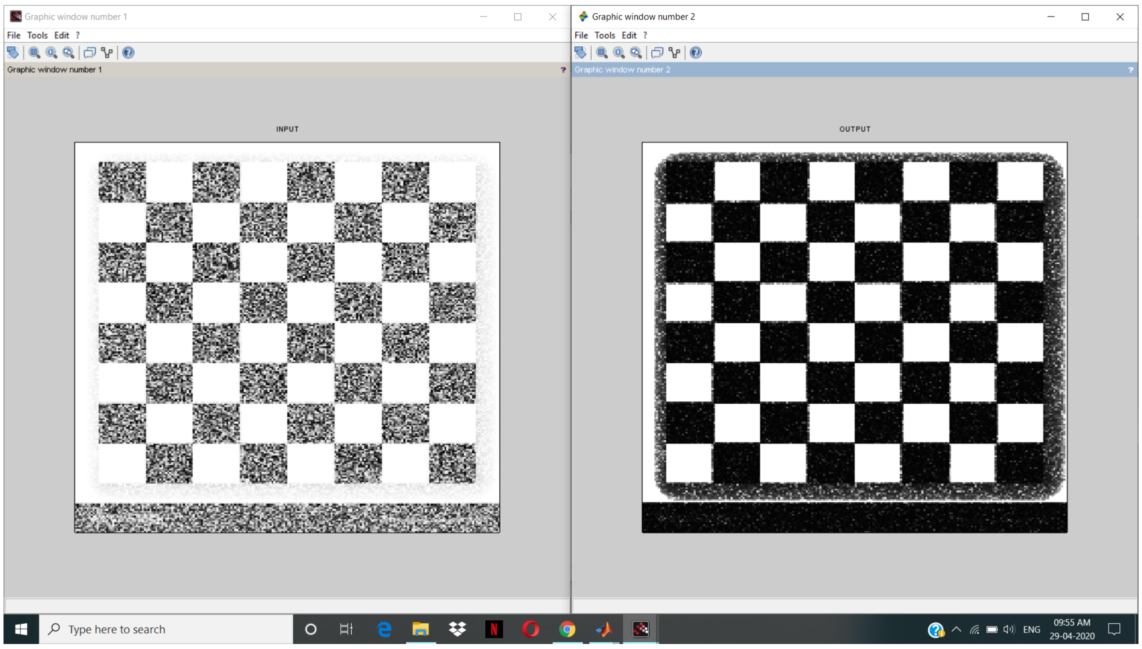
Listing 4.7: Recombination of the red blue green channel of the RGB images(RGB_recombiner.sci)

4.1.6 Versions

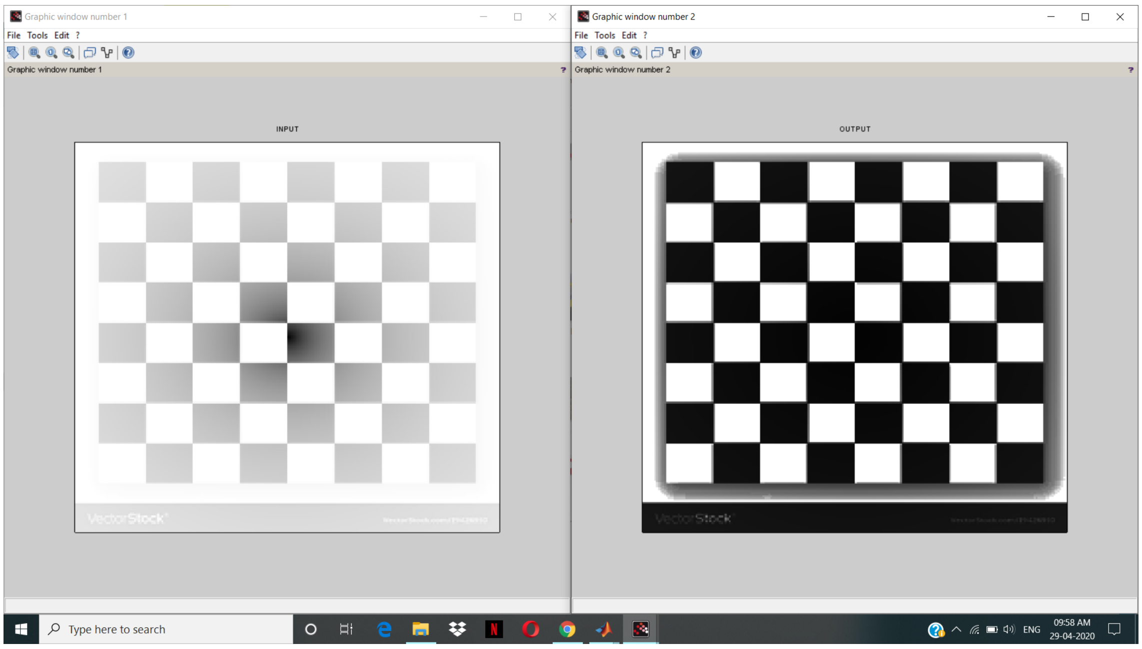
There are 3 versions which gets executed by executing the Case_study2.sce script which is comprised of namely:-

- **Version 1:**Retrieval of *gray-scale images* corrupted by the **Random Noise**.
- **Version 2:**Retrieval of *gray scale images* corrupted by the **Cloud Type Noise**.
- **Version 3:**Retrieval of *RGB images* corrupted by the **Cloud Type Noise**.

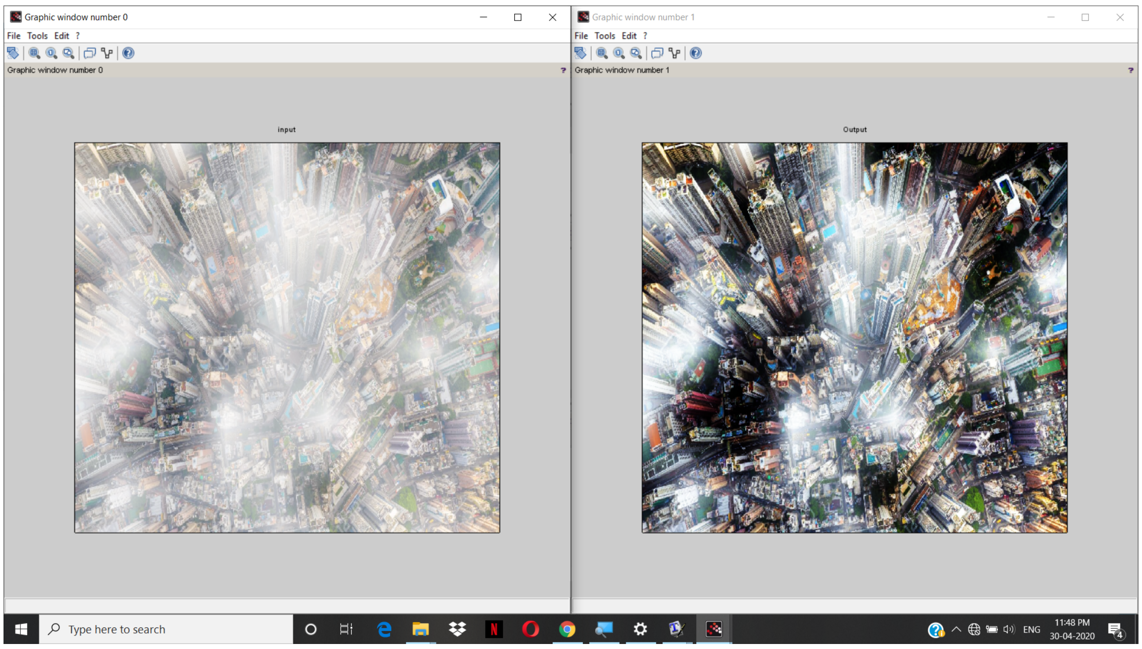
4.1.7 Output



Version 1 output



Version 2 output



Version 3 output

There are several other examples which is not shown here which can be seen during execution of the code or in the detailed documentation of the project.
More numbers of examples can included in *run_cloud.sci* file for further analysis of the algorithm.

Chapter 5

Xcos TBC

5.1 Abstract

Various Signals as well as Systems can be analyzed and can be realized using Xcos.Xcos Simulation can really help in resolving many issues in time varying system. It can play key role in understanding the causal system and the limitation of it which is hard or some times impossible to realize while or simply not feasible in actual design.

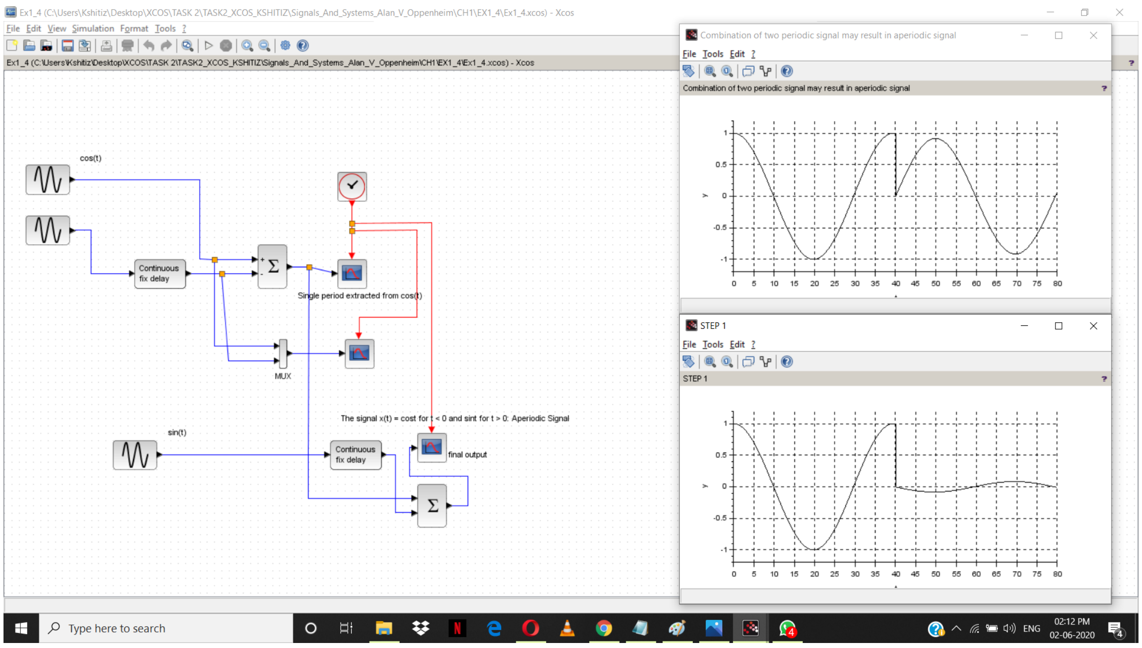
We have tried to cover as many block as possible (such as Scifunc Super_f block etc) in these 5 books to provide better documentation regarding the usage of each block.

5.2 List of Books

5.2.1 Signal and System by **Alan V.Oppenheim Alan S. Willsky S.Hamid Nawab**

Total six(6) chapters have been coded from the book which cover 20 examples of different types.There can be many other examples which can be solved from the book but here we have only considered those examples whose solution is already provided by the author but not in coded form.

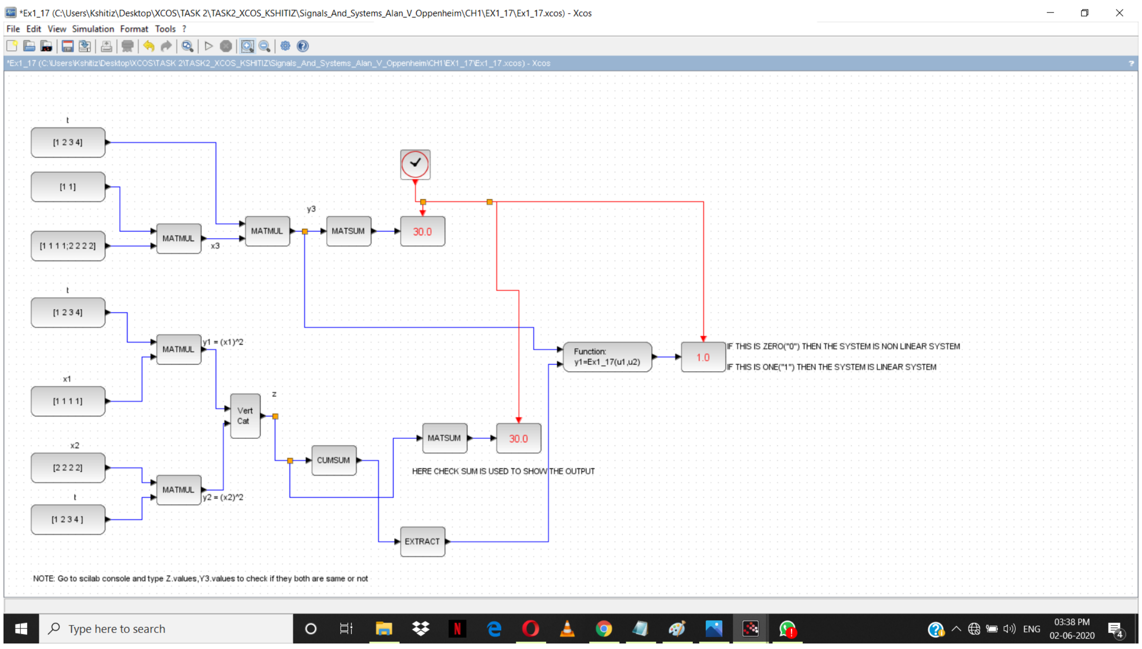
Here we have enlisted some coded examples to get the zest of the Xcos usability using various examples the first one shows the method of generation of an aperiodic signal/matrix operations/System response- poles and zeros.


cos(t)
sin(t)
Continuous fix delay
MUX
Single period extracted from cos(t)
final output
Combination of two periodic signal may result in aperiodic signal
STEP 1

Ex1_4 (C:\Users\Kshitiz\Desktop\XCOS\TASK 2\TASK2_XCOS_KSHITIZ\Signals_And_Systems_Alan_V_Oppenheim\CH1\EX1_4\Ex1_4.xcos) - Xcos

The signal x(t) = cost for t < 0 and sint for t > 0: Aperiodic Signal

Generation of the Aperiodic signal.



*Ex1_17 (C:\Users\Kshitiz\Desktop\XCOS\TASK 2\TASK2_XCOS_KSHITIZ\Signals_And_Systems_Alan_V_Oppenheim\CH1\EX1_17\Ex1_17.xcos) - Xcos

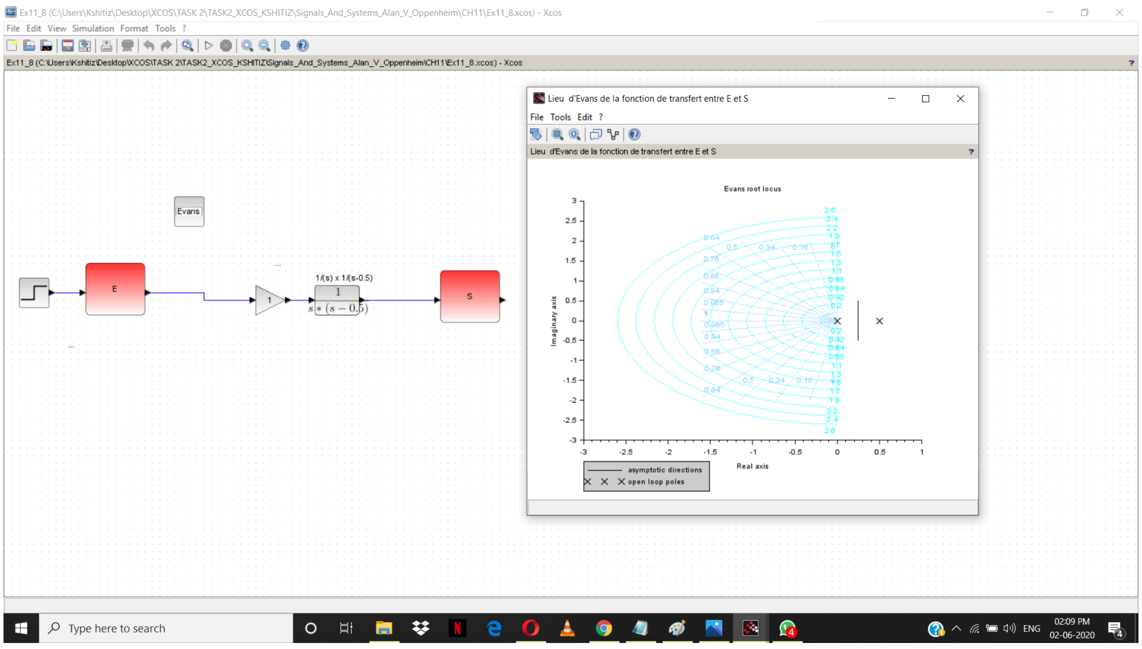
IF THIS IS ZERO("0") THEN THE SYSTEM IS NON LINEAR SYSTEM

IF THIS IS ONE("1") THEN THE SYSTEM IS LINEAR SYSTEM

HERE CHECK SUM IS USED TO SHOW THE OUTPUT

NOTE: Go to scilab console and type Z.values,Y3.values to check if they both are same or not

Various Matrix Operations as well as logical operation can also be done to realize a system.


Ex11_8 (C:\Users\Kshitiz\Desktop\XCOS\TASK 2\TASK2_XCOS_KSHITIZ\Signals_And_Systems_Alan_V_Oppenheim\CH11\Ex11_8.xcos) - Xcos
File Edit View Simulation Format Tools ?
Evans
E
1
1/(s) x 1/(s-0.5)
S
Lieu d'Evans de la fonction de transfert entre E et S
File Tools Edit ?
Evans root locus
Imaginary axis
Real axis
asymptotic directions
open loop poles
Type here to search
02:09 PM
02-06-2020

Various system response related graphs can be plotted using CPGE toolbox.

5.2.2 Integrated Electronics: Analog And Digital Circuits and Systems by J. Millman And C. C. Halkias

Total ten(10) chapters have been coded from the book which cover 20 examples of different types.There can be many other examples which can be solved from the book but here we have considered only those examples whose solution is already provided by the author but not in coded form.

5.2.3 Semiconductor Physics and Devices Basic Principles by D.A Neamen

Total eleven(11) chapters have been coded from the book which cover 22 examples of different types.There can be many other examples which can be solved from the book but here we have considered only those examples whose solution is already provided by the author but not in coded form.

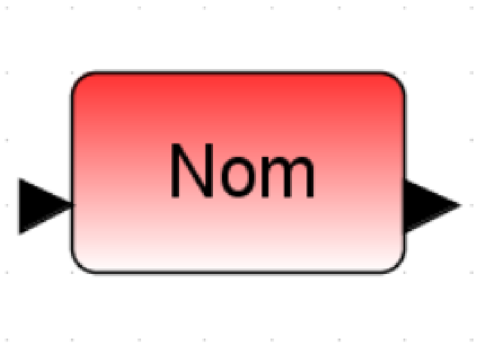
5.2.4 Principles of Electronic Communication Systems by L.E Frenzel

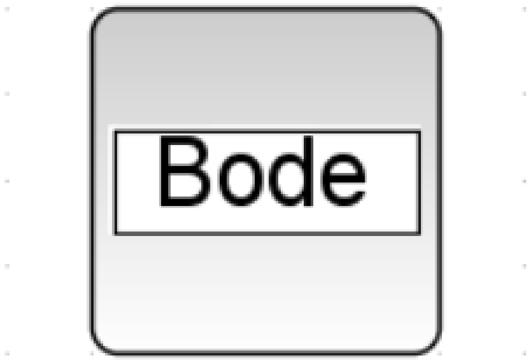
Total eleven(11) chapters have been coded from the book which cover 20 examples of different types.There can be many other examples which can be solved from the book but here we have considered only those examples whose solution is already provided by the author but not in coded form.

5.2.5 Quantum Physics Of Atoms, Molecules, Solids, Nuclei And Particles by Eisberg And R. Resnick

Total thirteen(13) chapters have been coded from the book which cover 20 examples of different types.There can be many other examples which can be solved from the book but here we have considered only those examples whose solution is already provided by the author but not in coded form.

5.2.6 Major Highlights of the CPGE Toolbox





The block on left allows to identify some physical quantities of the diagram, in order to plot frequency diagrams between a physical quantity in input and a physical quantity in output.
Whereas the block on right allows to plot frequency diagrams in Bode, Black or Nyquist plans, in open or closed loop, for any linear function

5.2.7 Problems of using CPGE toolbox on Xcos on Cloud

The Xcos on Cloud does not support the toolboxes currently and so you cannot test the code made from those toolboxes on the Xcos-on-Cloud. But you can use it on the native Scilab installed on your PC.

5.2.8 Other Useful Xcos Toolboxes

- Coselica
- Arduino
- MicroDAQ
- Scilab Code Generator

Chapter 6

Useful Links

6.0.1 Case Study 1 - full documentation

https://drive.google.com/file/d/1TwKVoSSShWMcmxTfcn1zoELV687zx1rK/view?usp=sharing

6.0.2 Case Study 2 - full documentation

https://drive.google.com/file/d/1axWOgGQ1Uf3YOMUjILr3jNXAGsVWEfbl/view?usp=sharing

6.0.3 Contacts

Kshitiz

email kshitiz205@iiitmanipur.ac.in

Reference

- https://www.scilab.org/
- https://cloud.scilab.in/
- http://xcos.fossee.in
- https://atoms.scilab.org/toolboxes/CPGE
- https://atoms.scilab.org/toolboxes/IPCV