Department of Electrical and Electronic Engineering

Dedan Kimathi University of Technology

Signals and Communication I Laboratory Manual

Ciira Maina, PhD

October 2015

Contents

1 Introduction **2**
1.1 Safety Information 2

2 System Setup **2**
2.1 Software Installation 2

3 Laboratory 1: Introduction to Python **3**
3.1 Objectives 3
3.2 Equipment 3
3.3 Background 3
3.3.1 Starting Python 3
3.3.2 Simple calculations 3
3.3.3 Packages 4
3.3.4 Vectors and Matrices 4
3.3.5 Plotting Graphs 5
3.3.6 Flow control 5
3.3.7 Scripts and Functions 6
3.4 Procedure 6

A Laboratory Report Format **8**

1 Introduction

This laboratory manual is designed to accompany the Signals and Communication I course at Dedan Kimathi University of Technology. It describes laboratory exercises aimed at integrating the Raspberry Pi into the curriculum. The development of these laboratory exercises was made possible by a generous grant from The Kenya Education Network (KENET).

The Raspberry Pi which retails at approximately $30 was developed with the aim of improving computing education by providing access to an affordable computer with most of the capabilities of a modern computer. It is about the size of a credit card (see Figure 1) and as we will see in the labs described here, it can perform a number of important DSP tasks.

Figure 1: The Raspberry Pi. A Kenyan bank card is shown for comparison.

1.1 Safety Information

A number of safety precautions must be observed when handling and operating the Raspberry Pi. You will have received a copy before starting the labs.

2 System Setup

The computing environment used in this lab is the raspbian Debian Wheezy operating system. This operating system is quite similar to Ubuntu. The labs will be based on Python, which is very useful open source high level language gaining popularity in engineering applications. The exercises have been tested on Python version 2.7.3.

2.1 Software Installation

To run the labs, Python must be installed. In addition we will need the following libraries

- Numpy - for numerical computations
- Scipy - for audio file manipulation.
- Matplotlib - for plotting

These software has already been installed on the devices that will be issued to you.

3 Laboratory 1: Introduction to Python

3.1 Objectives

The objectives of this laboratory are:

1. To introduce the Python programming language.
2. To demonstrate vector manipulation operations.
3. To demonstrate flow control in Python.
4. To demonstrate definition of functions in Python.

3.2 Equipment

For this lab you will need the following (If an item is not provided the student is expected to come with their own to the lab):

1. Raspberry Pi Model B+
2. Mini USB cable (not provided)
3. Ear phones or speakers (not provided)

3.3 Background

Python is a high level programming language with much of the capabilities of Matlab. It is very useful for numerical computations and this makes it ideal for prototyping DSP applications. Also, by making use of a number of packages, you have access to a number of useful functions. This lab is based on the tutorial found here `https://docs.python.org/2/tutorial/`

3.3.1 Starting Python

1. To start the Python interpreter, type `python` in the command line.
2. Python will start and display the promt

```
Python 2.7.3 (default, Feb 27 2014, 19:58:35)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

3.3.2 Simple calculations

You can use Python as a simple calculator.

1. To add two numbers `>>> 2+2`
2. To subtract two numbers `>>> 8-2`
3. To multiply two numbers `>>> 2*2`
4. To raise one number to a power `>>> 2**2`

5. We can define variables and perform mathematical operations on them for example

   >>> N=2

   >>> N*N

3.3.3 Packages

There are a number of important packages

1. Numpy and Scipy provide useful functions for scientific work `http://docs.scipy.org/doc/numpy/reference/`

2. Matplotlib is useful for making plots `http://matplotlib.org/`

3. For example to compute the sine of an angle, we import numpy and use the sine function defined in the numpy package. We also use the value of π defined in numpy

   ```
   import numpy
   numpy.sin(numpy.pi)
   ```

   For example to compute and print $\sin(2\pi)$ type

   ```
   numpy.sin(2*numpy.pi)
   ```

   To avoid too much typing we can assign a short name to an imported module. For example

   ```
   import numpy as np
   np.sin(np.pi)
   ```

4. To find out what a function does and how it works, use help. For example help(np.sin).

3.3.4 Vectors and Matrices

Vectors and matrices are very important in signals and communication applications. Many signals we will manipulate will be represented by vectors.

1. We can define a vector by placing its elements within square brackets and using the array function in the numpy package.

   ```
   a=np.array([1, 2, 3, 4])
   ```

2. We can determine the first element of the vector by typing a[0]. In Python the indices are zero based like C++.

3. We can determine the dimensions of the vector using the function np.shape(a)

4. We can create M-by-N matrices using the functions np.ones and np.zeros. For example

   ```
   np.ones((2,2))
   ```

5. We can take the element-wise multiplication of two vectors using the * operator, for example a*a.

6. We can also take the dot product of two vectors by using the dot function in numpy np.dot(a,a)

7. Two compatible matrices can also be multiplied using the dot function in numpy, for example np.dot(np.ones((5,5)),np.ones((5,1)))

3.3.5 Plotting Graphs

Python can be used to produce good quality graphs for reports and publications. For this, we use the matplotlib package. For basic plots we import the pyplot module in matplotlib

```
import matplotlib.pyplot as plt
```

1. To plot the simple function $y = x$ in the interval $0 \leq x \leq 1$,
   (a) Generate the vector x=np.linspace(0,1,100);
   (b) Define y, y=x;
   (c) To generate the figure type plt.plot(x,y)
   (d) To display the figure type plt.show()
2. Explore the use of plt.title, plt.xlabel, plt.ylabel, plt.legend to label the axes and include titles for graphs.
3. The figure can be saved using plt.savefig.

3.3.6 Flow control

Here we discuss for loops, while loops and if else statements.

1. The syntax of a for loop is:

   ```
   for index in list:
       expression
   ```

   where we iterate over the items in the list. These could be numbers, words, characters etc. For example to simply print the numbers zero to nine we can write

   ```
   for i in [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]:
       print i
   ```

   The second line must be indented. The convention is to use 4 spaces. In place of the list [0, 1, 2, 3, 4, 5, 6, 7, 8, 9], we could use range(10). Explore the use of the range function using help.

2. The syntax of a while loop is:

   ```
   while (condition):
       expression
   ```

   For example to print the numbers one to ten we would write

   ```
   i=1
   while i<=10:
       print i
       i+=1
   ```

3. The syntax for if else statements is

```
if (condition):
    expression
elif (condition):
    expression
else:
    expression
```

3.3.7 Scripts and Functions

Sometimes it is more convinient to write a sequence of commands in a text file than to type them one after another in the command line. We can write a module with a number of functions and import this module when we want to use these functions. These files a saved with a .py extension

The code is written in a file using the following format. Lines after the function declaration must be indented by 4 spaces.

```
def function_name (input1,input2,...):
    Code....
    Code...
    return output1,Output2,...
```

For example we could have a file powers.py in which functions to compute the square and cube of a number are defined.

```
def my_square(x):
    ans=x*x
    return ans

def my_cube(x):
    ans=x*x*x
    return ans
```

We could then import the powers module and use the functions defined in it

```
>>> import powers
>>> powers.my_cube(3)
27
```

3.4 Procedure

1. Sample the function $x(t) = \sin(2\pi t)$ at 100Hz and plot the function for $-2 \leq t \leq 2$. What is the period, T_0, of $x(t)$? What is the sampling period T_s?

2. In continuous time, the power of a periodic signal is given by

$$P_x = \frac{1}{T_0} \int_0^{T_0} |x(t)|^2 dt$$

   (a) What is the power of $x(t)$

(b) Show that we can estimate P_x using the samples $x[n]$ as

$$P_x \approx \frac{1}{T_0} \sum_{n=1}^{N} x^2[n] T_s$$

(c) Write a for loop to estimate P_x. How does your value compare with the value computed theoretically.

3. Write a module with functions capable of plotting the following signals

   (a) Unit step

   (b) Unit ramp

4. Using the module above plot the following signals

   (a) $u(t) - 2u(t-2) + u(t-3)$ where $u(t)$ is the unit step

   (b) $r(t) - 2r(t-2) + r(t-3)$ where $r(t)$ is the unit ramp

A Laboratory Report Format

For each of the laboratory exercises in this manual, you will be required to had in a report. Each report should have the following sections:

1. Title page:

   The title page should include

   - Course code and title
   - Title of the lab
   - Name and registration number of the student
   - Date

2. Introduction:

   In the introduction you should indicate the objectives of the laboratory and give any necessary background information such as relevant theory.

3. Procedure:

   Here you should describe in detail the steps carried out in the lab. The details should allow someone else to reproduce your work. You can include short code segments here but code listings should be placed in the appendix.

4. Results and Discussion:

   Here you should describe all the results obtained and compare them with what was theoretically expected. Data should be presented appropriately using tables and figure which are clearly labelled

5. Conclusion:

   Summarize the laboratory report and indicate whether the objectives of the lab were achieved.

6. References:

   Include any references using the IEEE format `http://www.ieee.org/documents/ieeecitationref.pdf`.

7. Appendix:

   Include any additional information such as detailed schematics and code listings here.