User Manual for PolyHarmonics

Nolan Driessen

August 31, 2015

Contents

1 Introduction

This document is designed to assist with installation, setup and upkeep of PolyHarmonics on Python version 2.7.

2 Installation

The following section provides an overview of what needs to be installed before PolyHarmonics will function.

2.1 Python

PolyHarmonics is designed to function on Python version 2.7, as such Python must be installed and updated to this version in order for the software to run properly.

2.2 Required Packages

The following packages do not come with Python 2.7 by default and must be installed separately.

- NumPy: Provides support for large multi-dimensional arrays and many high level mathematical functions.
- SciPy: Builds on NumPy with modules for optimization and many commonly used functions in scientific computing.
- Matplotlib: A Python 2D plotting library.
- Pywt: Used for wavelet transforms, also known as PyWavelets.
- npTDMS: Required in order to read TDMS files.

2.3 Installation Steps

The following method requires pip which comes preinstalled with Python version 2.7.9 and beyond.

- From the command line type the following command: pip install [package name]
- To verify the package was properly installed, open an Idle Python GUI and enter the command: import [package name]
- If no error is returned the packages have been properly installed. However if an import error is given, please refer to section 2.3.1 for further instructions.

2.3.1 Detailed Windows Installation Steps

In the event the steps outlined in section 2.3 fail follow these steps:

- From a command line enter the command: pip install update
- From either a Python shell or command line running Python run the following command: import pip; print pip.pep425tags.get_supported()

  Take note of the first entry as this will be used to install the proper modules for your system.

- From the following link download each required package according to the first entry returned from the previous step. Please note this website is unofficial.
  http://www.lfd.uci.edu/~gohlke/pythonlibs/

  For example, if the first entry returned from the python command was ('cp27', 'none', 'win32') you would want the NumPy the file called "numpy1.9.2+mklcp27nonewin32.whl". Download the proper version of each required module and place it within the Python27/Tools/Scripts folder.

- Navigate a command window to the Python/Tools/scripts folder and run the command:
  pip install your-package.whl
  Where your-package is the name of the file previously downloaded.

3 Using PolyHarmonics

3.1 Running

Once the specified modules have been properly installed PolyHarmonics can be run by opening the file main.py and choosing run.
Before running, the file config.txt must be modified to contain proper input. This file is located in the same directory as main.py. Open config.txt and edit the start frequency, stop frequency and step frequency as well as the Main Directory entries. The system will use these entries as input and give all output in the location specified by Main Directory.

4 Likely Changes

PolyHarmonics has been developed in order to make likely changes easily. When making a change, first refer to the Module Guide in order to identify which module will contain this change.

4.0.1 New Functions

In some cases, a new function will need to be created in order to accommodate a change. If this is the case, within the proper module create a new function and identify which input will be required from other modules. Using the Get function of that specific data you can receive the input from a separate module. Once the required input has been gathered the body of the function can be built according to the problem being solved. Finally, after the function is complete main.py must be modified to call this function, have main.py call the function using dot notation, module.function().

4.0.2 Modifying Existing Functions

In the case an existing function must be modified the steps will be similar to creating a new function. First identify the module that the change should be made in. If a function already

exists but must be modified, identify if the required input is already within the module. If necessary remove unnecessary Get functions, or add the new required ones. Finally modify the function accordingly, no changes should need to be made to main.py or any other module if the likely change is well contained.

5 Testing

This section describes running the test cases for PolyHarmonics, including setup and expected behaviour.

5.1 System Tests

There is one system test created at the moment. In order to run the test, ensure that the config.txt file is setup such that:
Start Frequency: 100
Stop Frequency: 1000
Step Frequency: 100
Main Directory: standard_test_cases/input 1
One additional package is required for the system testing, download Pillow using the steps outlined in section 2.3.1. Once the parameters are set, open Testing.py and run it. The system will run main.py and execute one test case, which compares the created text files and graphs to the expected output. If the system either errors or fails the program is not running as expected and must be fixed before being used for analysis.

5.2 Unit Tests

Unit testing tests each module on its own, rather than the whole system at once. The unit tests do not require any modification of config.txt. In order to run the unit tests, open unitTest.py and run it. This will execute 6 tests, and indicate if any error or fail.

5.3 Python Version 3

In order to convert the current Python 2.7 code into Python 3 code open a command line, navigate to the folder containing the PolyHarmonics modules and run the following command:
python C:/Python27/Tools/scripts/2to3.py -w [module to convert]
Where [module to convert] will be replaced by one of the files, such as main.py. This will have to be done for each module, and the result will have a .bak backup file and the Python 3 version of the converted code.