Python & Qt,
Powerful tools for technical Computing

90th S&V Symposium – Nov 3 – 7th, 2019
Atlanta, Ga

Vince Grillo
Dynamic Environments, AI Solutions, KSC LSP
Structural Dynamics

Python Overview

Python History and Design:

- Python is an interpreted (non-compiled), high-level, Open Source, general purpose programming language created in the late 1980s and released in 1991 by Guido van Rossum at the National Research Institute for Mathematics and Computer Science, (Centrum Wiskunde & Informatica – CWI) Amsterdam, Netherlands.
- Python was originally created as a successor to ABC programming language which is an imperative programming language meaning Python uses statements that change a program’s state like “Try” some block of code or “if variable is True” in describing how a program operates.
- Python supports Object Oriented programming and is “Dynamically Typed” meaning objects and extensions defined in the code like variable types are automatically loaded at runtime and do not need to be compiled.
- Garbage collection or memory management is handled automatically by the interpreter.
- Python interpreters are available for many operating platforms including: Windows, Linux and MacOS. Code written in Python is naturally platform independent.
- NASA LSP currently uses anaconda Python from Anaconda Inc.

Python programming structures

- Modules and Imports
  - Libraries created in Python to perform a wide range of functions in different disciplines.
- Data types:
  - Strings, Integers, Floats, Complex……etc.
- Data Structures: Tuples, lists, dictionaries, Sets,…..etc.
  - Tuples, ordered sequence of data that is immutable (can't be changed).
  - Lists, similar to Tuples but are mutable or can be changed.
  - Dictionaries, unordered key-value pairs for saving variables and storing data.
- Numpy – Numerical Python
  - Mathematical functions including multi-dimensional array structures.
- Scipy – Scientific Python
  - Set of scientific functions including signal processing.
- Matplotlib – Data plotting and manipulation
- Pandas – DataFrames and spreadsheet data
  - Powerful tools for tabular data manipulation.
- h5py – Hierarchical Data format for input and output.
  - Efficient tools for reading and writing data especially large data files.
- pyYeti – Set of customized Structural Dynamics Tools written in Python.
- System & OS Interfaces – sys and os, python interface to current os.

Qt Graphical Language Overview

- Qt is a graphical user interface (GUI) programming language that provides for rapid GUI development in a python environment well beyond Tcl/Tk GUI language.
- Qt was originally developed in Finland at Trolltech, which is now The Qt Company by Haavard Nord & Eirik Chambe-Eng.
- Qt was ported to Python by Phil Thompson as a separate python module pyqt and is currently at Qt version 5.0.
- Qt uses object oriented programming interfaces or Widgets, Layouts and Canvases with a unique application of communication between interfaces using Signals and Slots.
- Widgets include: Main Menus, Spin Boxes, Drop down Menus, Icons, push buttons, radio buttons, check boxes ...etc.

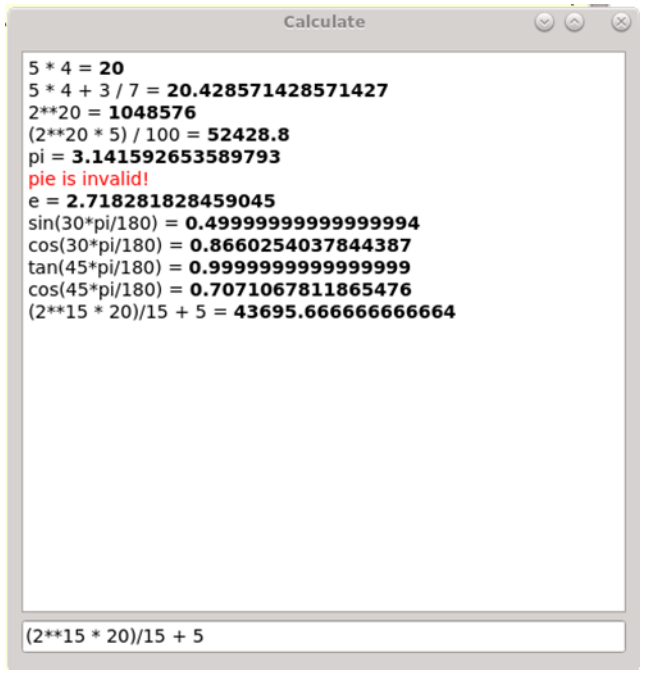
Qt Programming Structures

- PyQt5 QtGui
- PyQt5 QtWidgets, QIcons
- PyQt5 QTableView, QTextEdit
- PyQt5 QtCore
- PyQt5.QtWidgets : QVBoxLayout, QHBoxLayout
- Matplotlib Backends for Qt5:
  - Figure Canvas : matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg as FigureCanvas
  - Navigation Toolbar : matplotlib.backends.backend_qt5agg import NavigationToolbar2QT as NavigationToolbar
  - Core Interfaces: matplotlib.backends.backend_qt5agg import QtCore, QtGui
- Many other libraries within Qt5 Core module including program signals: From PyQt5.QtCore import pyqtSignal

Sample Python Qt code example

- Interactive GUI command line calculator – 40-lines, no-compilation.

```
from __future__ import division
import sys
from math import *
import scipy
import numpy as np
import PyQt5.QtCore
import PyQt5.QtGui
from PyQt5.QtCore import *
from PyQt5.QtGui import *
from PyQt5 import QtWidgets


class Form(QtWidgets.QDialog):

    def __init__(self, parent=None):
        super(Form, self).__init__(parent)
        self.browser = QtWidgets.QTextBrowser()
        self.lineedit = QtWidgets.QLineEdit("Type an expression and press Enter")
        self.lineedit.selectAll()
        layout = QtWidgets.QVBoxLayout()
        layout.addWidget(self.browser)
        layout.addWidget(self.lineedit)
        self.setLayout(layout)
        self.lineedit.setFocus()
        self.lineedit.returnPressed.connect(self.updateUi)
        self.setWindowTitle("Calculate")


    def updateUi(self):
        try:
            text = self.lineedit.text()
            self.browser.append("{} = <b>{}</b>".format(text,
                                        eval(text)))
        except:
            self.browser.append("<font color=red>{} is invalid!</font>"
                                        .format(text))


app = QtWidgets.QApplication(sys.argv)
form = Form()
form.show()
app.exec_()
```

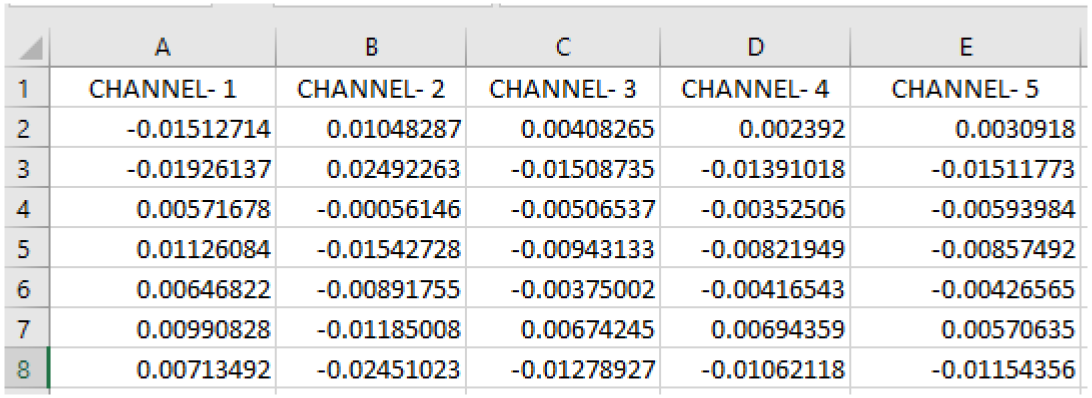


```
5 * 4 = 20
5 * 4 + 3 / 7 = 20.428571428571427
2**20 = 1048576
(2**20 * 5) / 100 = 52428.8
pi = 3.141592653589793
pie is invalid!
e = 2.718281828459045
sin(30*pi/180) = 0.49999999999999994
cos(30*pi/180) = 0.8660254037844387
tan(45*pi/180) = 0.9999999999999999
cos(45*pi/180) = 0.7071067811865476
(2**15 * 20)/15 + 5 = 43695.666666666664
```

```
(2**15 * 20)/15 + 5
```

Vibration Lab test data example and Matlab/Excel problems

- Shaker Vibration Data Acquisition(DAS) systems don't necessarily format data that will be compatible with Matlab and Windows.
  - Sample data set - 5-channels, over 1M rows:

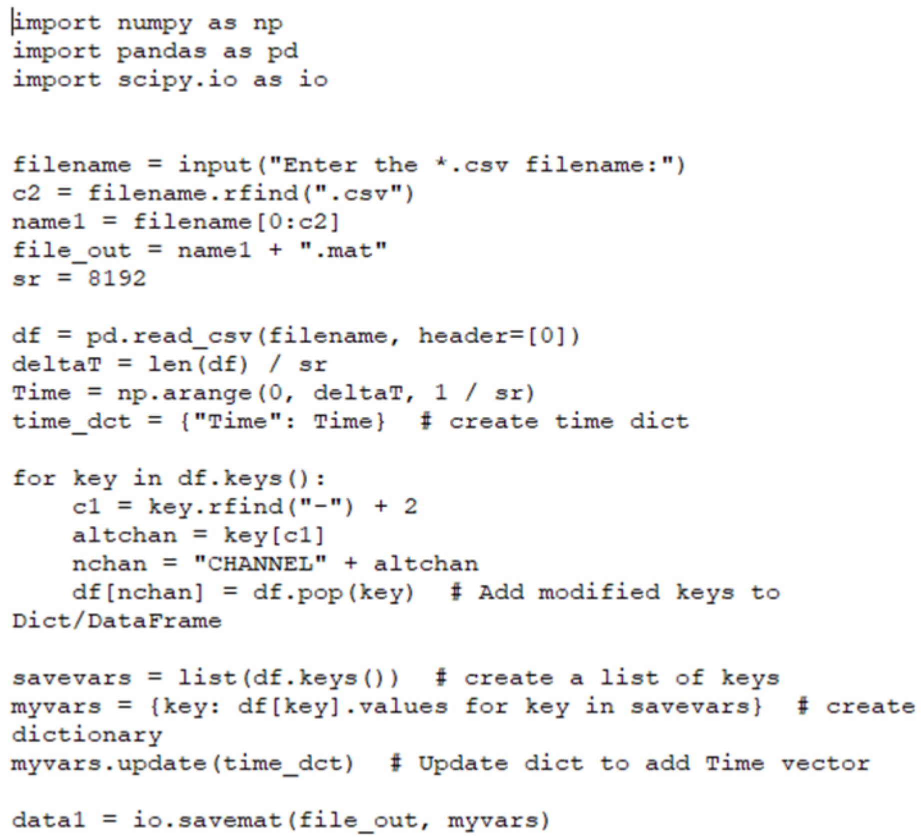


	A	B	C	D	E
1	CHANNEL- 1	CHANNEL- 2	CHANNEL- 3	CHANNEL- 4	CHANNEL- 5
2	-0.01512714	0.01048287	0.00408265	0.002392	0.0030918
3	-0.01926137	0.02492263	-0.01508735	-0.01391018	-0.01511773
4	0.00571678	-0.00056146	-0.00506537	-0.00352506	-0.00593984
5	0.01126084	-0.01542728	-0.00943133	-0.00821949	-0.00857492
6	0.00646822	-0.00891755	-0.00375002	-0.00416543	-0.00426565
7	0.00990828	-0.01185008	0.00674245	0.00694359	0.00570635
8	0.00713492	-0.02451023	-0.01278927	-0.01062118	-0.01154356

  - Column names with dashes and spaces not compatible with Matlab – [CHANNEL-1]
  - CSV files well over 1,000,000 rows are readable by excel, but difficult and slow to manipulate.
  - The X-Axis_Control.csv data set in this example is 1,048,544 rows.
  - No time vector created by DAS, time vector cannot be easily created in excel.

Solution to previous Matlab/Excel issues

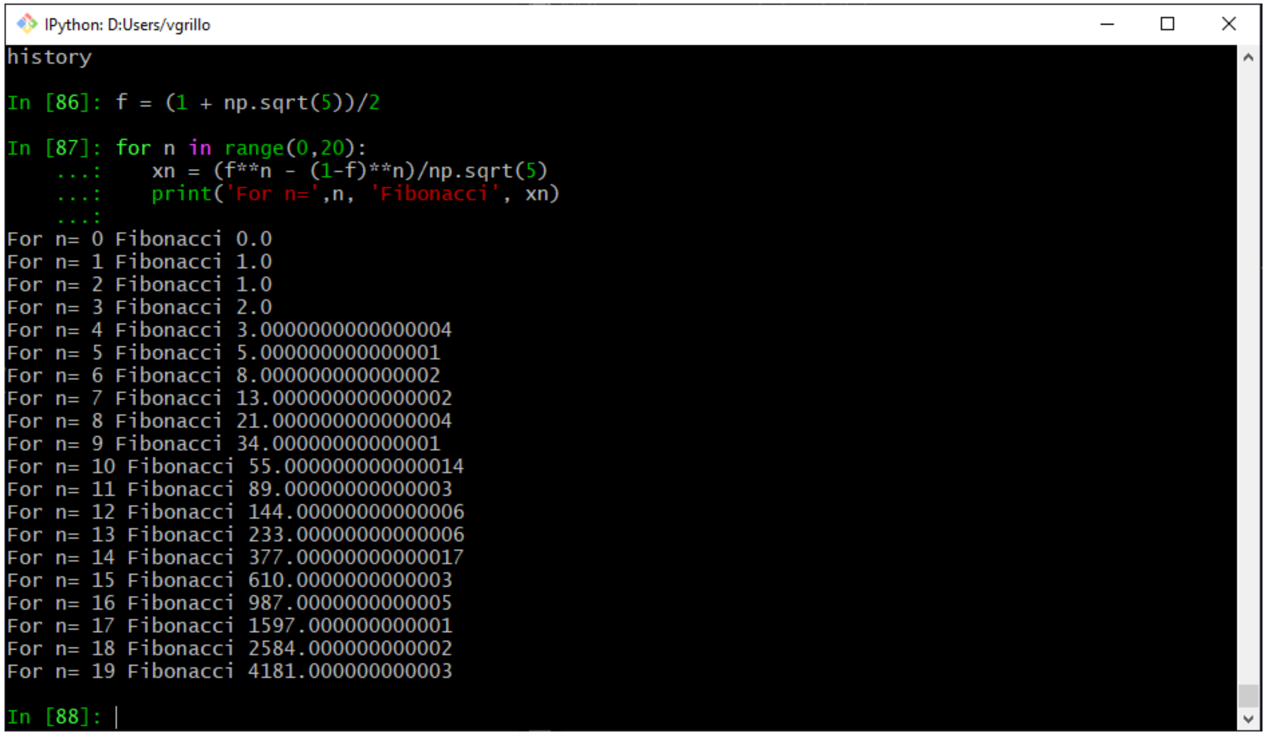
- Python Code below illustrates how variables read in from ‘.csv’ file can easily be re-assigned to variable names compatible with Matlab and saved in Matlab format.
- Note: Additional Import statements calling tools mentioned in prior slides.



```
import numpy as np
import pandas as pd
import scipy.io as io


filename = input("Enter the *.csv filename:")
c2 = filename.rfind(".csv")
name1 = filename[0:c2]
file_out = name1 + ".mat"
sr = 8192

df = pd.read_csv(filename, header=[0])
deltaT = len(df) / sr
Time = np.arange(0, deltaT, 1 / sr)
time_dct = {"Time": Time}  # create time dict

for key in df.keys():
    c1 = key.rfind("-") + 2
    altchan = key[c1]
    nchan = "CHANNEL" + altchan
    df[nchan] = df.pop(key)  # Add modified keys to
Dict/DataFrame

savevars = list(df.keys())  # create a list of keys
myvars = {key: df[key].values for key in savevars}  # create
dictionary
myvars.update(time_dct)  # Update dict to add Time vector

data1 = io.savemat(file_out, myvars)
```

Python Shell – Git Hub Desktop

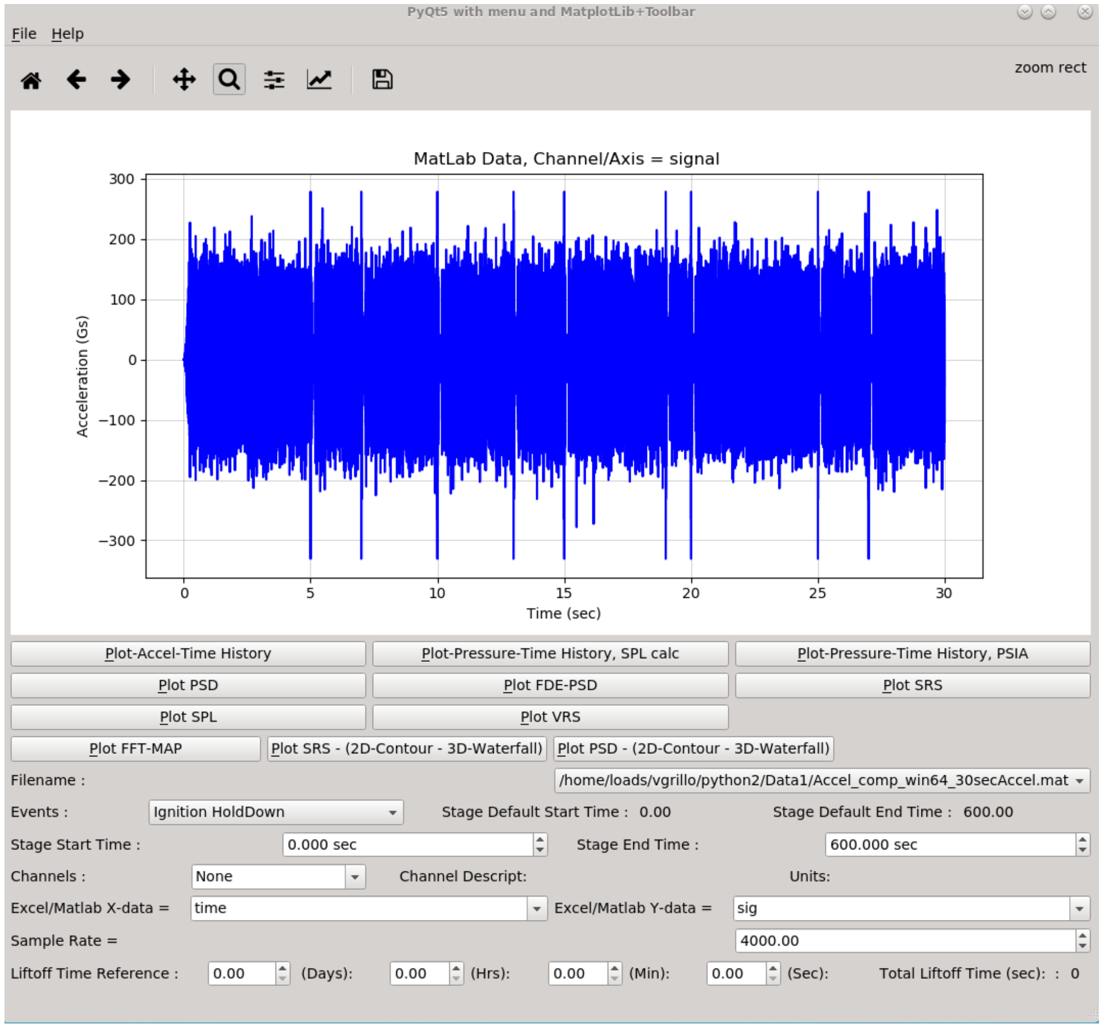
- Windows based shell for running Python & Ipython
- Linux, Xwindows shell like behavior in a windows environment.
- Git for Windows - https://gitforwindows.org/
- Example : Fibonacci number generator in 4-lines.



```
history

In [86]: f = (1 + np.sqrt(5))/2

In [87]: for n in range(0,20):
    ...:     xn = (f**n - (1-f)**n)/np.sqrt(5)
    ...:     print('For n=',n, 'Fibonacci', xn)
    ...:
For n= 0 Fibonacci 0.0
For n= 1 Fibonacci 1.0
For n= 2 Fibonacci 1.0
For n= 3 Fibonacci 2.0
For n= 4 Fibonacci 3.0000000000000004
For n= 5 Fibonacci 5.000000000000001
For n= 6 Fibonacci 8.000000000000002
For n= 7 Fibonacci 13.000000000000002
For n= 8 Fibonacci 21.000000000000004
For n= 9 Fibonacci 34.00000000000001
For n= 10 Fibonacci 55.000000000000014
For n= 11 Fibonacci 89.00000000000003
For n= 12 Fibonacci 144.00000000000006
For n= 13 Fibonacci 233.00000000000006
For n= 14 Fibonacci 377.00000000000017
For n= 15 Fibonacci 610.0000000000003
For n= 16 Fibonacci 987.0000000000005
For n= 17 Fibonacci 1597.000000000001
For n= 18 Fibonacci 2584.000000000002
For n= 19 Fibonacci 4181.000000000003

In [88]:
```

Structural Dynamics Graphical User Interface (GUI)

- Custom GUI written in PyQt based on Structural Dynamics tools written in python – pyYeti.
- Start Demo – multiplotgui


PyQt5 with menu and MatplotLib+Toolbar
File Help
zoom rect
MatLab Data, Channel/Axis = signal
Acceleration (Gs)
Time (sec)
300
200
100
0
−100
−200
−300
0
5
10
15
20
25
30
Plot-Accel-Time History
Plot-Pressure-Time History, SPL calc
Plot-Pressure-Time History, PSIA
Plot PSD
Plot FDE-PSD
Plot SRS
Plot SPL
Plot VRS
Plot FFT-MAP
Plot SRS - (2D-Contour - 3D-Waterfall)
Plot PSD - (2D-Contour - 3D-Waterfall)

Filename : /home/loads/vgrillo/python2/Data1/Accel_comp_win64_30secAccel.mat

Events : Ignition HoldDown   Stage Default Start Time : 0.00   Stage Default End Time : 600.00

Stage Start Time : 0.000 sec   Stage End Time : 600.000 sec

Channels : None   Channel Descript: Units:

Excel/Matlab X-data = time   Excel/Matlab Y-data = sig

Sample Rate = 4000.00

Liftoff Time Reference : 0.00 (Days): 0.00 (Hrs): 0.00 (Min): 0.00 (Sec):   Total Liftoff Time (sec): : 0

Conclusion

- Python & PyQt5 provide a powerful set of tools for technical computing and visualization.
  - Complex problems in Structural Dynamics as well as other disciplines can be easily interacted with and visualized.
  - Python addresses a broad array of technical disciplines, as of Nov. 2018, there are over 63,000 modules for import into python.
  - Python is command line interpreted and can be easily written and executed on a variety of operating systems.
  - PyQt is a powerful language for building GUIs quickly to interact with data and provide real-time results.

- Any Questions?