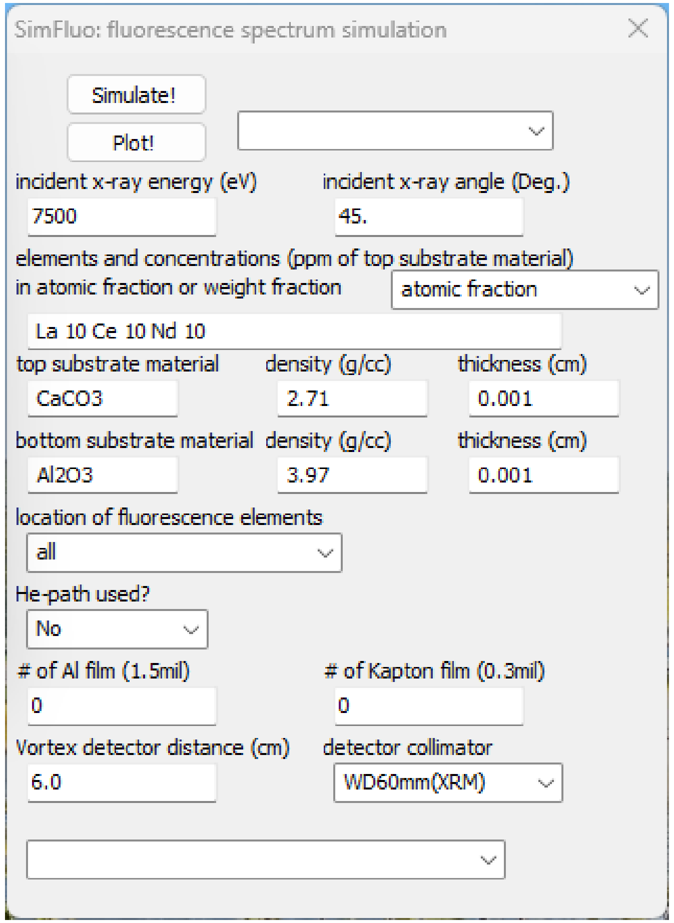
7/9: python-GUI-1

- File for GUI:
  - fluo_panel.py
- Files for calculation:
  - elam, fl1f2chantler, readf1f2a, fluo_elem, fluo_det.py

fluo_panel.py

fluo_panel.py



SimFluo: fluorescence spectrum simulation

incident x-ray energy (eV): 7500

incident x-ray angle (Deg.): 45.

elements and concentrations (ppm of top substrate material) in atomic fraction or weight fraction: atomic fraction

La 10 Ce 10 Nd 10

top substrate material: CaCO3 | density (g/cc): 2.71 | thickness (cm): 0.001

bottom substrate material: Al2O3 | density (g/cc): 3.97 | thickness (cm): 0.001

location of fluorescence elements: all

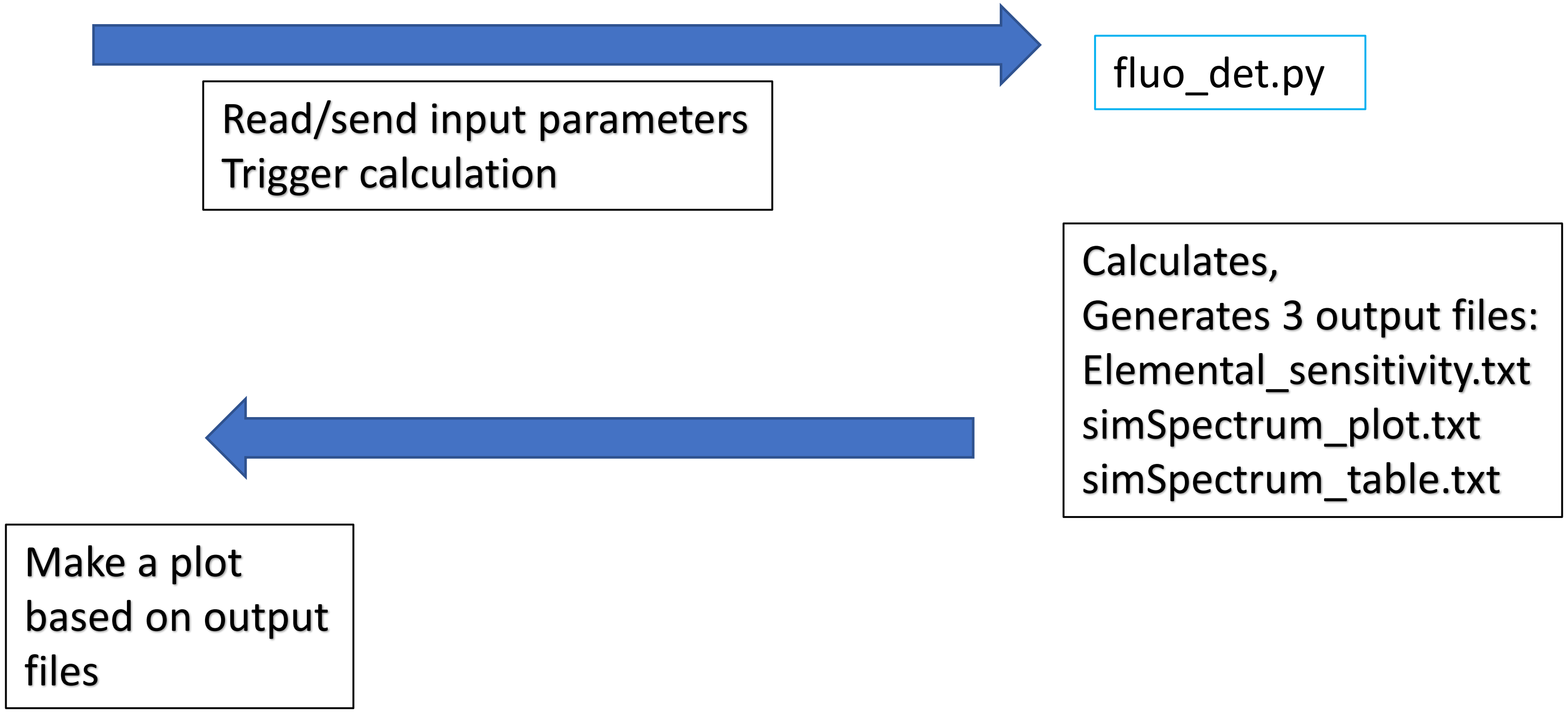
He-path used?: No

# of Al film (1.5mil): 0

# of Kapton film (0.3mil): 0

Vortex detector distance (cm): 6.0

detector collimator: WD60mm(XRM)



Read/send input parameters
Trigger calculation

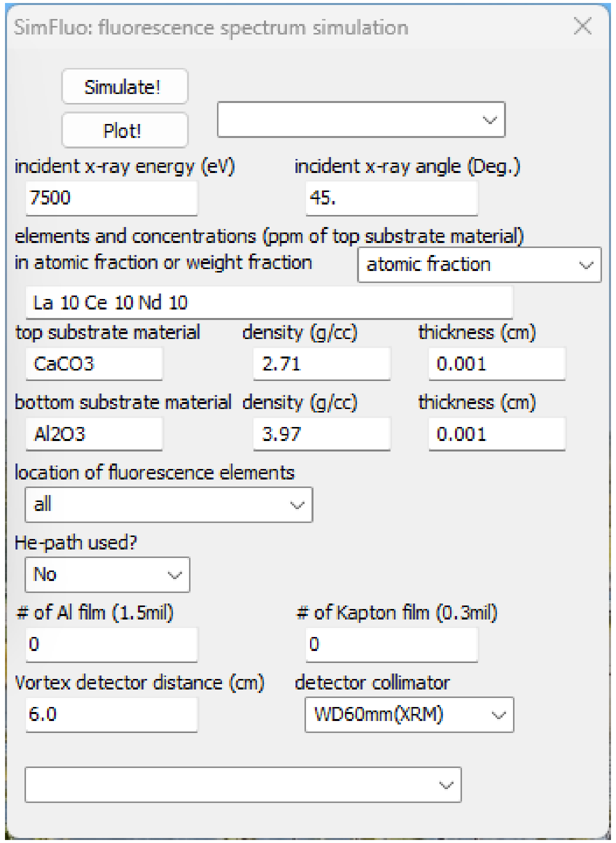
fluo_det.py

Calculates,
Generates 3 output files:
Elemental_sensitivity.txt
simSpectrum_plot.txt
simSpectrum_table.txt

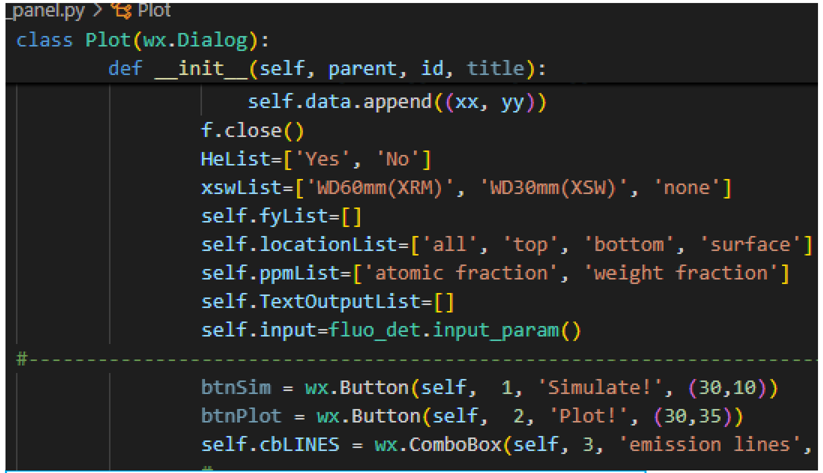
Make a plot based on output files

7/9: python-GUI-2

- Details of GUI (using wxPython)
    - Layout of options, functions for calculation, linking function to option

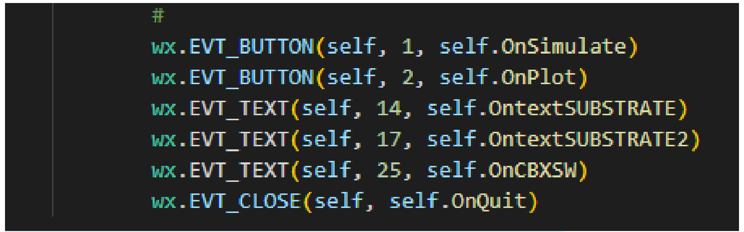


incident x-ray energy (eV): 7500
incident x-ray angle (Deg.): 45.
elements and concentrations (ppm of top substrate material) in atomic fraction or weight fraction: atomic fraction
La 10 Ce 10 Nd 10
top substrate material: CaCO3 density (g/cc): 2.71 thickness (cm): 0.001
bottom substrate material: Al2O3 density (g/cc): 3.97 thickness (cm): 0.001
location of fluorescence elements: all
He-path used?: No
# of Al film (1.5mil): 0
# of Kapton film (0.3mil): 0
Vortex detector distance (cm): 6.0
detector collimator: WD60mm(XRM)



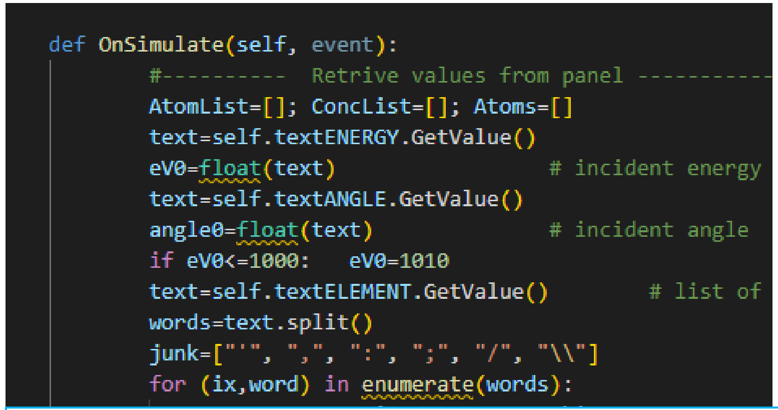
```
_panel.py > Plot
class Plot(wx.Dialog):
        def __init__(self, parent, id, title):
                        self.data.append((xx, yy))
                f.close()
                HeList=['Yes', 'No']
                xswList=['WD60mm(XRM)', 'WD30mm(XSW)', 'none']
                self.fyList=[]
                self.locationList=['all', 'top', 'bottom', 'surface']
                self.ppmList=['atomic fraction', 'weight fraction']
                self.TextOutputList=[]
                self.input=fluo_det.input_param()
#-----------------------------------------------------------------------
                btnSim = wx.Button(self,  1, 'Simulate!', (30,10))
                btnPlot = wx.Button(self,  2, 'Plot!', (30,35))
                self.cbLINES = wx.ComboBox(self, 3, 'emission lines',
```

Layout: making input windows



```
        #
        wx.EVT_BUTTON(self, 1, self.OnSimulate)
        wx.EVT_BUTTON(self, 2, self.OnPlot)
        wx.EVT_TEXT(self, 14, self.OntextSUBSTRATE)
        wx.EVT_TEXT(self, 17, self.OntextSUBSTRATE2)
        wx.EVT_TEXT(self, 25, self.OnCBXSW)
        wx.EVT_CLOSE(self, self.OnQuit)
```

Linking option to function:

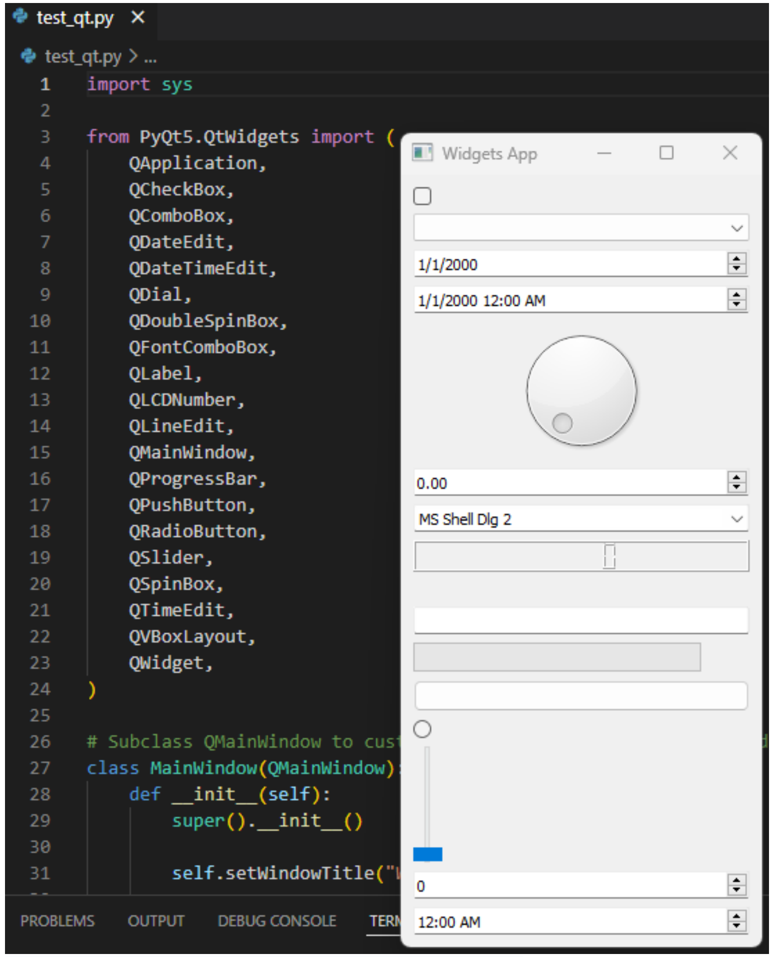


```
def OnSimulate(self, event):
        #----------  Retrive values from panel ----------
        AtomList=[]; ConcList=[]; Atoms=[]
        text=self.textENERGY.GetValue()
        eV0=float(text)                  # incident energy
        text=self.textANGLE.GetValue()
        angle0=float(text)               # incident angle
        if eV0<=1000:   eV0=1010
        text=self.textELEMENT.GetValue()        # list of
        words=text.split()
        junk=["'", ",", ":", ";", "/", "\\"]
        for (ix,word) in enumerate(words):
```

Function: what to do when “Simulate” is pressed.

7/9: python-GUI-3

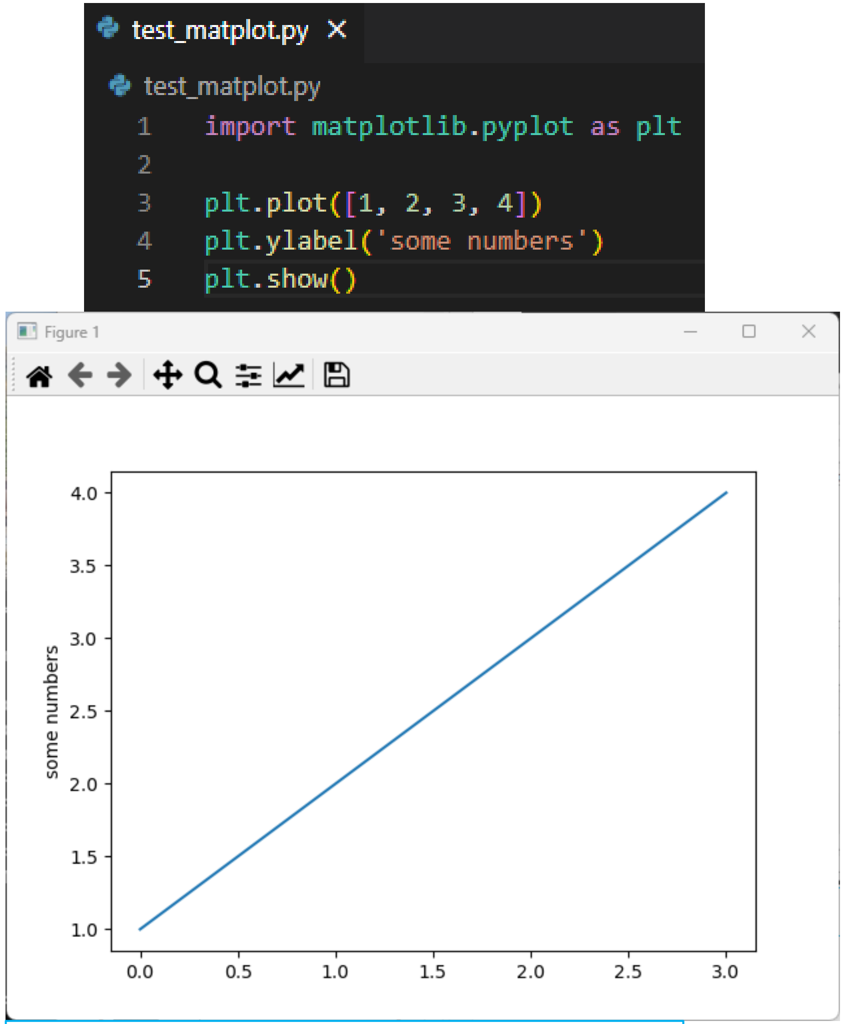
- PyQt5: Try tutorial for make layout.
- Matplotlib: Try tutorial for plotting.



```python
import sys

from PyQt5.QtWidgets import (
    QApplication,
    QCheckBox,
    QComboBox,
    QDateEdit,
    QDateTimeEdit,
    QDial,
    QDoubleSpinBox,
    QFontComboBox,
    QLabel,
    QLCDNumber,
    QLineEdit,
    QMainWindow,
    QProgressBar,
    QPushButton,
    QRadioButton,
    QSlider,
    QSpinBox,
    QTimeEdit,
    QVBoxLayout,
    QWidget,
)

# Subclass QMainWindow to cust
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        self.setWindowTitle("
```

```
1/1/2000
1/1/2000 12:00 AM
0.00
MS Shell Dlg 2
0
12:00 AM
```

test_qt.py: from tutorial



```python
import matplotlib.pyplot as plt

plt.plot([1, 2, 3, 4])
plt.ylabel('some numbers')
plt.show()
```

test_matplotlib.py: from tutorial