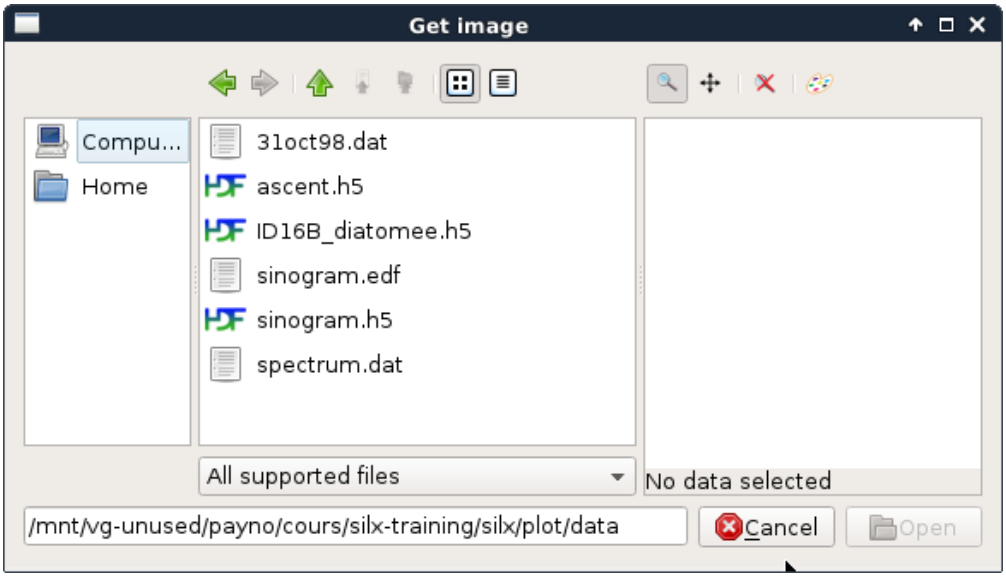
normalizationExample

March 16, 2018

```
In [ ]: %gui qt
```

Same exercise, (data, normalization function) as io but with interaction

1 Create two functions



ImageDialog

- `getFlatfield`: to select the flatfield
- `getDark`: to select the dark

To do this use the ImageFileDialog class

- see doc: http://www.silx.org/doc/silx/latest/modules/gui/dialog/imagefiledialog.html
- related example: examples/fileDialog.py

```
In [ ]: from silx.gui.dialog.ImageFileDialog import ImageFileDialog
```

```
In [ ]: import os
        def getImage(title=None):
            dialog = ImageFileDialog()
            dialog.setWindowTitle(title or 'Get image')
            dir_ = 'data'
            assert os.path.isdir(dir_)
            dialog.setDirectory(dir_)
            if not dialog.exec_():
                return None
            else:
                return dialog.selectedImage()
```

```
In [ ]: def getFlatfield():
            return getImage('Select flatfield')
```

```
In [ ]: def getDark():
            return getImage('Select dark')
```

2 Create a function to correct an image from flatfield and dark

take as input an image, dark and flatfield. Return the normalized image

```
In [ ]: def normalized(image, flatfield, dark):
            return (image - dark) / (flatfield - dark)
```

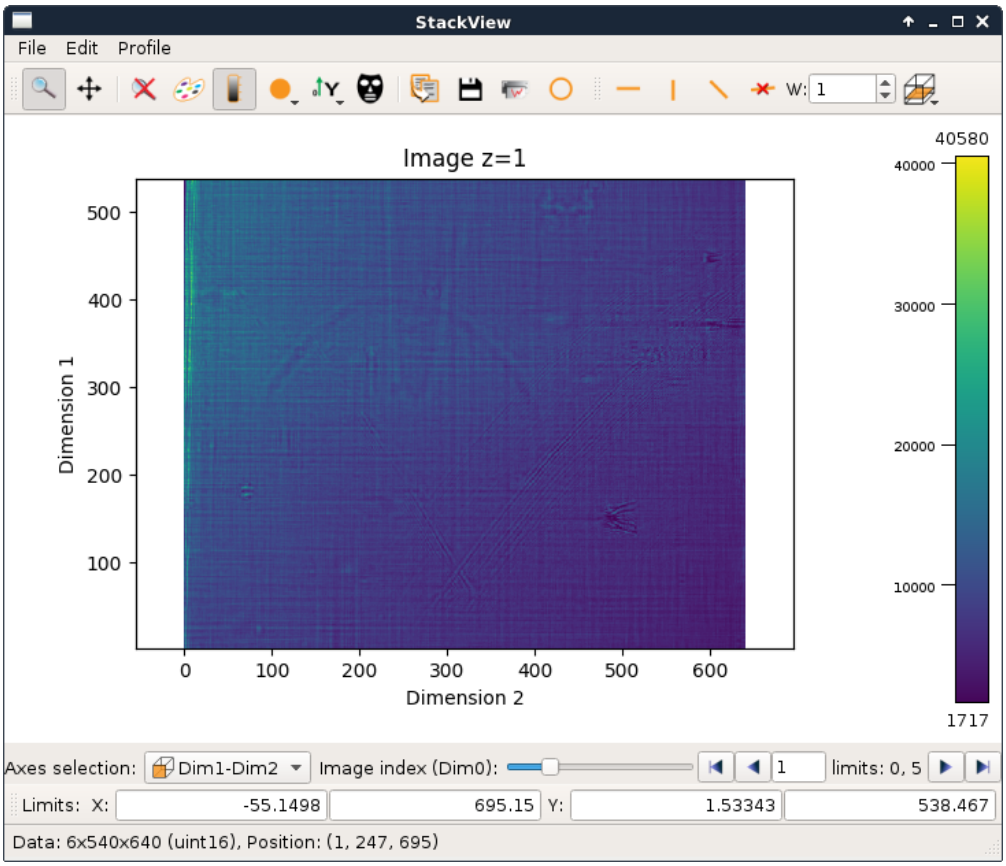
3 Select an image and display it raw and normalized

```
In [ ]: from silx.gui.plot import Plot2D
        def showRawAndCorrected(raw, corrected):
            plot = Plot2D()
            plot.addImage(data=raw, replace=False, legend='raw')
            plot.addImage(data=corrected, replace=False,
                          legend='corrected', origin=(raw.shape[0], 0))
            plot.show()
```

```
In [ ]: image = getImage()
        flatfield = getFlatfield()
        dark = getDark()
        if image is not None and dark is not None and flatfield is not None:
            corrected = normalized(image=image,
                                   flatfield=flatfield,
                                   dark=dark)
            showRawAndCorrected(raw=image, corrected=corrected)
```

4 Add an action to apply the correction on a stack of image

Here is the sample code to plot the stack of image



ImageDialog

```
In [ ]: from silx.gui.plot.StackView import StackViewMainWindow
        import h5py
        import numpy

        dataFile = h5py.File('data/ID16B_diatomee.h5')

        mystack = dataFile['scan1']['instrument']['data'][...]

        sv = StackViewMainWindow()
        sv.setStack(mystack)
        sv.show()
```

Here is a function to apply the corection on the stack

```
In [ ]: def applyCorrection(images, flatfield, dark):
            correctedImgs = []
            for image in images:
                correctedImgs.append(normalized(image,
                                                flatfield=flatfield,
                                                dark=dark))
            return correctedImgs
```

To define an action: - heritate from PlotAction - redefine the triggered function - See the tutorial on how to add an action see: http://www.silx.org/doc/silx/dev/modules/gui/plot/actions/examples.html - you can also use the PlotAction tutorial.ipynb

```
In [ ]: from silx.gui.plot.actions import PlotAction
        class CorrectImageAction(PlotAction):
            """QAction applying the correction algorithm

            :param plot: :class:`.PlotWidget` instance on which to operate
            :param parent: See :class:`QAction`
            """
            def __init__(self, plot, sv, parent=None):
                PlotAction.__init__(self,
                                    plot,
                                    icon='shape-circle',
                                    text='process correction',
                                    tooltip='apply the correction algorithm',
                                    triggered=self._actionTriggered,
                                    parent=parent,
                                   checkable=False)
                self.stackViewer = sv

            def _actionTriggered(self):
                flatfield = getFlatfield()
                dark = getDark()
                if flatfield is None or dark is None:
```

```
            return
        newStack = applyCorrection(images=self.stackViewer.getStack()[0],
                                   flatfield=flatfield,
                                   dark=dark)
        self.stackViewer.setStack(newStack)
```

In [ ]:
```
sv = StackViewMainWindow()
sv.setStack(mystack)

toolBar = sv.getPlot().toolBar()
myaction = CorrectImageAction(sv.getPlot(), sv)
toolBar.addAction(myaction)
sv.show()
```