```
## PLOTING COMPLEX X-RAY ABSORPTION DATA ##
## Create an object named PFY to load XAS data with polygon ROI shape
PFY = PFYLoader()

## LOADING/ADDING/SUBSTRACTING 1-D/REDUCED DATA FROM A FILE ##
## Loads XES scans data from HDF5 file
PFY.load(config,'filename', 'detector', *args, **kwargs)
## *args = comma seperated list of scans to be plotted or added and then plotted

## Loads and sums XES scans data from HDF5 file
PFY.add(config,'filename', 'detector', *args, **kwargs)
## *args = comma seperated list of scans to be plotted or added and then plotted

## Loads and subtratcs XES scans data from HDF5 file
PFY.subtract(config,'filename', 'detector', *args, **kwargs)
## *args = s1, p1 -> The data from p1 is subtracted from s1
## *args = [s1, ..., sn], [p1, ..., pn] -> The sum of p1..pn is sub. from the sum s1...sn

## Loads and stitches non-overlapping regions
PFY.stitch(config,'filename', 'detector', *args, **kwargs)
## *args = comma seperated list of scans to be stitched
## NOTE: The the scans are appended in order, overlap discared

## Loads and subtract scan from all previously loaded scans
PFY.background(config,'filename', 'detector', *args, **kwargs)
## *args = s1 -> The scan to be subtracted from all previous load/add/subtract actions
## *args = [s1, ..., sn] -> The sum of scans s1..sn to be subtracted from all previous load/add/subtract actions

## REQUIRED VARIABLES ##
## config = RIXS               -> RIXS Endstation
## config = RSXS               -> RSXS Endstation
## filename = hdf5 file        -> Extension .h5 not needed
## detector                  -> sum of MCA type detector
## detector[Start:End]       -> sums all MCA data within emission energy range
## detector[S1:E1,S2,E2] -> sums all MCA data within energy range S1 to E1 (start) and S2 to E2 (end)
## NOTE: Simple math allowed with xes_stream with contstants and variables, i.e. +, -, /, *

## **kwargs ##
## norm = True                         -> Scales the data such that its range is 0 to 1.
## twin_y = True                       -> Adds these plots to a secondary scale
## xoffset = [(S1,P1),...,(SN,PN)]     -> Adjusts x-axis scale to map SN to PN
## xcoffset = value                    -> Shifts x-axis scale by a constant value
## yoffset = [(S1,P1),...,(SN,PN)]     -> Adjusts y-axis scale to map SN to PN
## ycoffset = value                    -> Shifts y-axis scale by a constant value
## grid = [start,stop,delta]         -> Change x-axis grid to be uniform
## savgol = (wind len, poly ord, derv) -> Smooths and takes derivative

## SET RANGE OF Y and X VALUES ##
PFY.xlim(min, max)
PFY.ylim(min, max)
## NOTE: These ranges will be preserved in the data export

## PLOTTING SCAN DATA ##
PFY.plot(**kwargs)

## **kwargs ##
## title = 'New Title of plot' -> Replaces default title with user defined
## xlabel = 'x-axis label'     -> Replaces default x-axis label with user defined
## ylabel = 'y-axis label'     -> Replaces default y-axis label with user defined
## plot_height = value         -> The plot height in points, default is 600
## plot_width = value          -> The plot width in points, default is 900
## norm = True                 -> Normalizes all the data between 0 and 1
## waterfall = offset          -> Normalizes as above and shifts each by the offset

## EXPORTING PLOT DATA ##
PFY.export('filename', **kwargs)

## REQUIRED VARIABLES ##
## filename = filename to be used for ASCII file, do not add extension
## NOTE: Data is exported as it displayed, only options in plotting methods are ignored.

## **kwargs ##
## split_files = True -> Saves each data stream with number appended to the filename
```

```
## Loading PFY data to remove elastic scattering and compare
PFY = PFYLoader()
PFY.load(RIXS, 'HDF5_Notebook', 'XES[380:385,380:425]/i0',25,26)
PFY.load(RIXS, 'HDF5_Notebook', 'XES[380:400]/i0',25)
PFY.load(RIXS, 'HDF5_Notebook', 'XES/i0',25)
PFY.plot()
PFY.export('PFY_Test')
```