PySaxs
A Python module and GUI for SAXS data treatment

Olivier Taché

Collaborative work with :
O. Spalla, A. Thill, D. Sen, D. Carrière, F. Testard

Outline

Context :

- CEA-LIONS
- Small Angle X-Rays Scattering
- SAXS at LIONS
- SAXS data treatment

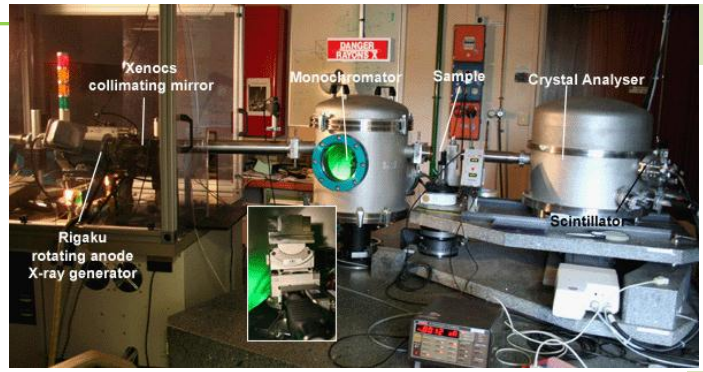
Python

What is pySAXS ?

User Interface : GuiSAXS

- plots
- data treatment
- fitting by models

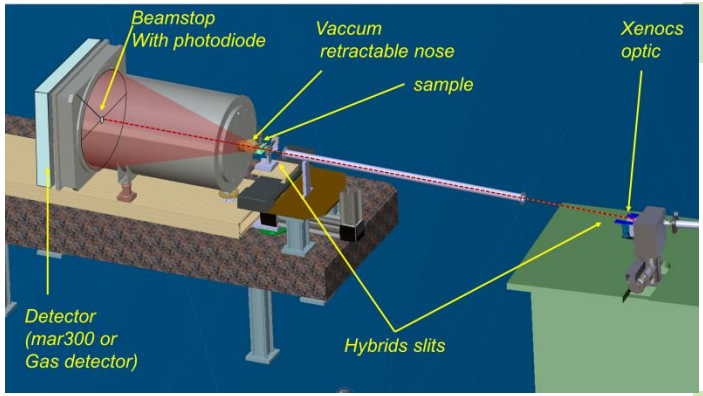
SAXS at LIONS : 3 experimentals setups



USAXS ultra small angles

q range : $2x10^{-4}$ to 10^{-1} A^{-1}

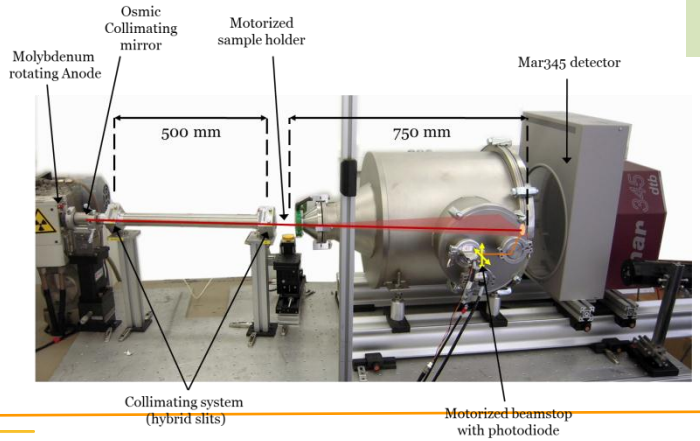
λ=0.154 nm
E=8keV
1D detector



SAXS

q range : $2x10^{-2}$ to $7x10^{-1}$ A^{-1}

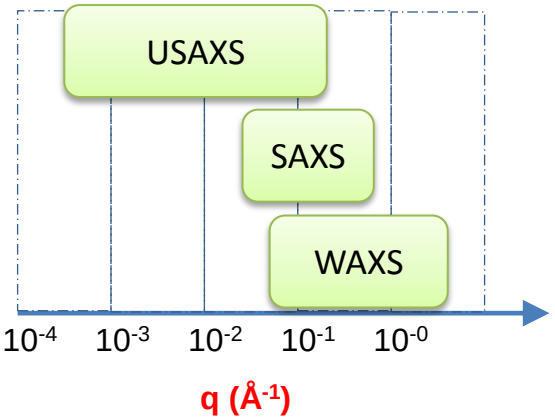
λ=0.154 nm
E=8 keV
2D detector



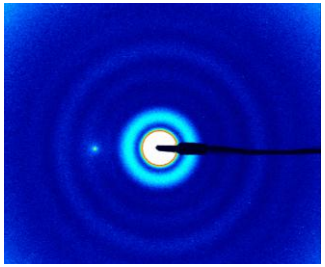
SAXS – WAXS (wide angles)

q range : $4x10^{-2}$ to 4 A^{-1}

λ=0.07 nm
E=17 keV
2D detector



SAXS Data treatment : what we have to do

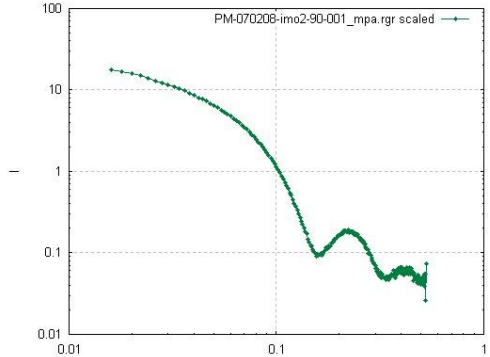


1- for Images : data reduction

Using ImageJ (Java !) a software that manage images (with ROI, LUT)
With geometrical corrections

1- for USAXS (1D) : deconvolution (beam is not perfect)

With specific code



2- scaling in absolute intensities (taking in acount experimental parameters)

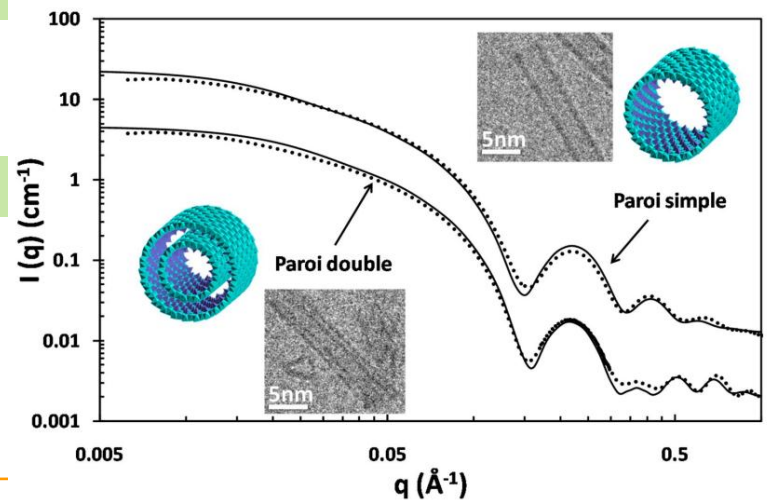
Very important if we want to compare datas from others experiments (synchrotron)
We can calculate Form factor and Structure Factor

3- merging datas and substract background or solvant (ie water)

Merging datas with different scales (qrange or dq)

4- compare with predefined models

minimization for finding sample parameters
Home made models
or with source code we can check…



5- non automatic data processing for calculating form factor, structure,…

With source code optimized and tested

→ Home made software → PySAXS

Other softwares ?

Other SAXS data treatment softwares :

- Sasfit : for neutron, C language
- SOLEIL : foxtrot (integrated with the hardware)
- Igor routines (not free, code source)
- Matlab routines (not free, code source)
- BioXtas (python with a similar wxPython GUI)
- Glatter (not free, code source)

With python, researchers can **validate** and modify the source code

With GuiSAXS, standard users can analyze **easily** datas

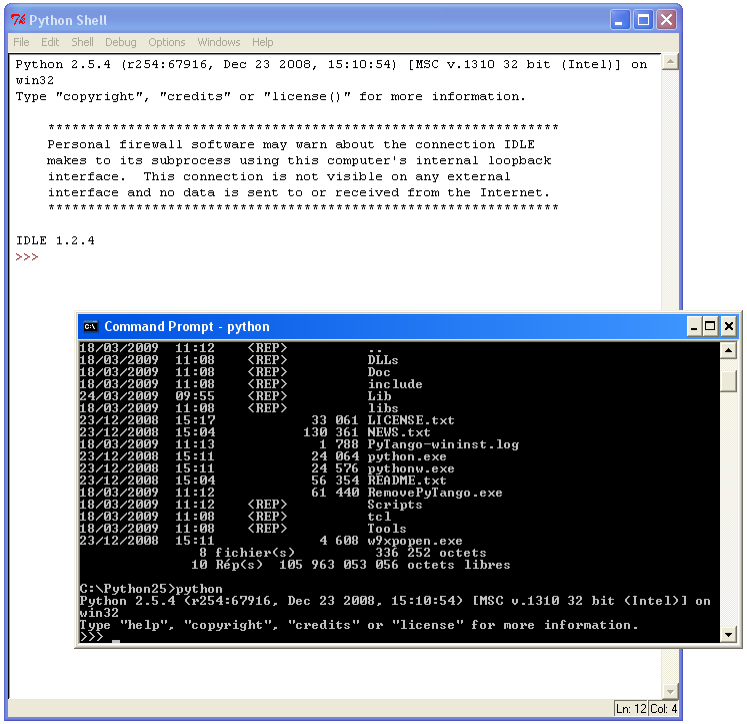
Python ?

- Langage de programmation « simple » qui permet de se concentrer sur l'application scientifique et pas la syntaxe
- Python a été conçu pour être un langage lisible. Il vise à être visuellement épuré. L'indentation est obligatoire.
- Orienté Objet, donc évolué
- modulaire, donc évolutif
- Utilisation dans de nombreux contextes
- Interface avec d'autres langages (Fortran, C,…)
- Portable (utilisable sous unix, mac, windows,…)
- Interfaçage avec de nombreuses librairies graphiques
- Nombreuses librairies scientifiques
- Open Source et gratuit

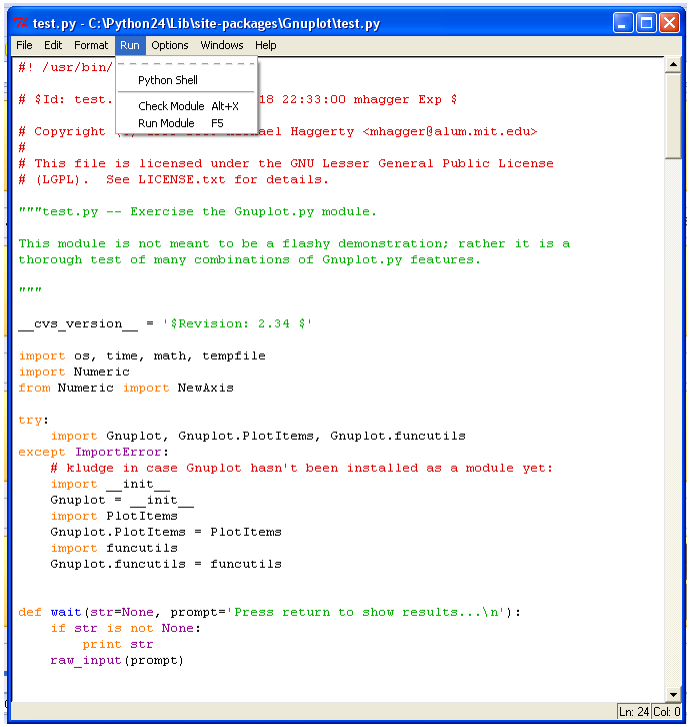
- Utilisation en ligne de commande, ou en « programme »

Utilisation de Python

1- Interpréteur Python



2- Exécution dans l'interpréteur



3- Exécution « directe » (dos, linux)


```
Python.exe monprogramme.py
```

Python scientifique: Les tracés avec Matplot

Gnuplot

Matplotlib

Scipy

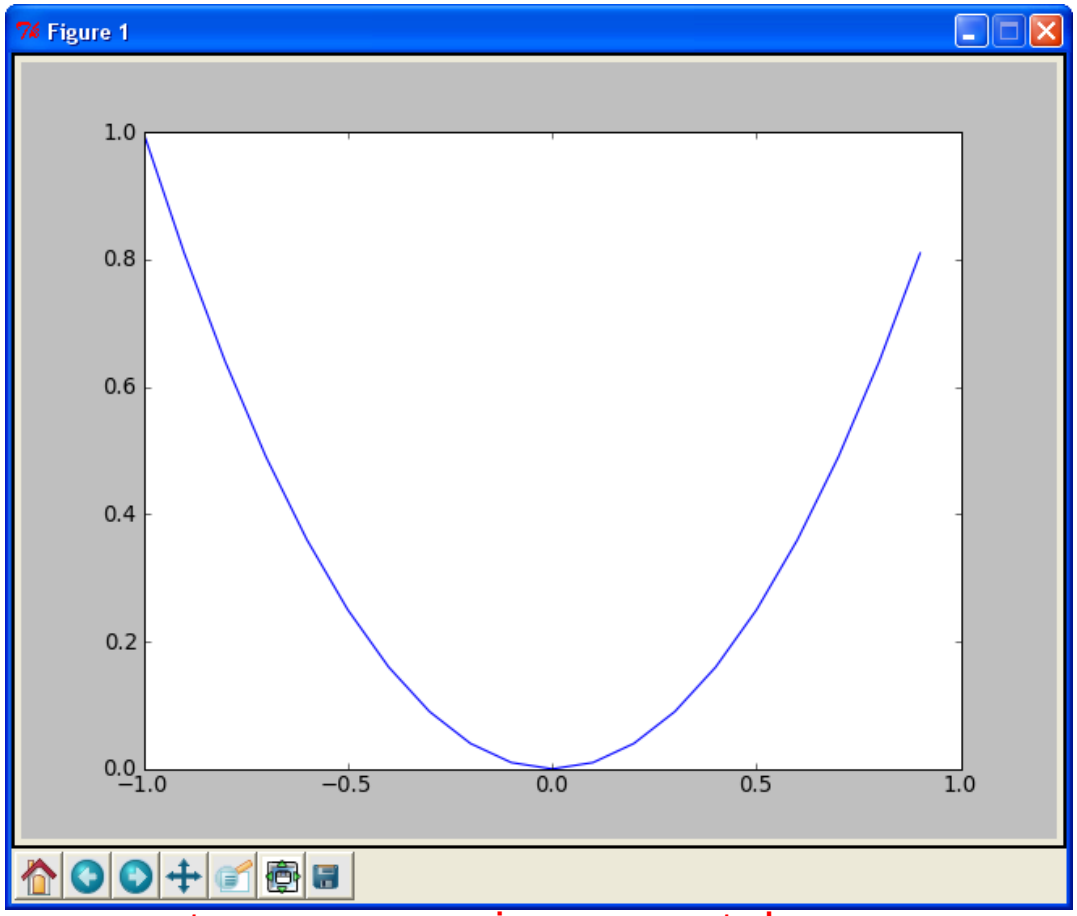
Numpy

Python

```
import pylab
from  numpy import *
x=arange(-1.0,1,0.1)
y=x*x

fig=pylab.figure()
axes=fig.gca()
axes.plot(x, y, linewidth=1.0)

fig.show()
```



retrouver un environnement de développement interactif similaire à ceux de MATLAB ou IDL

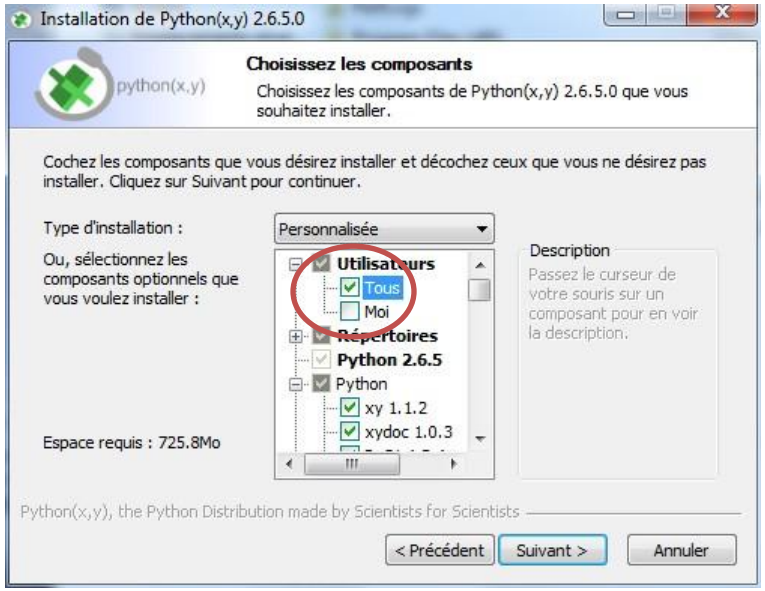
Python scientifique :Distribution Python(x,y)

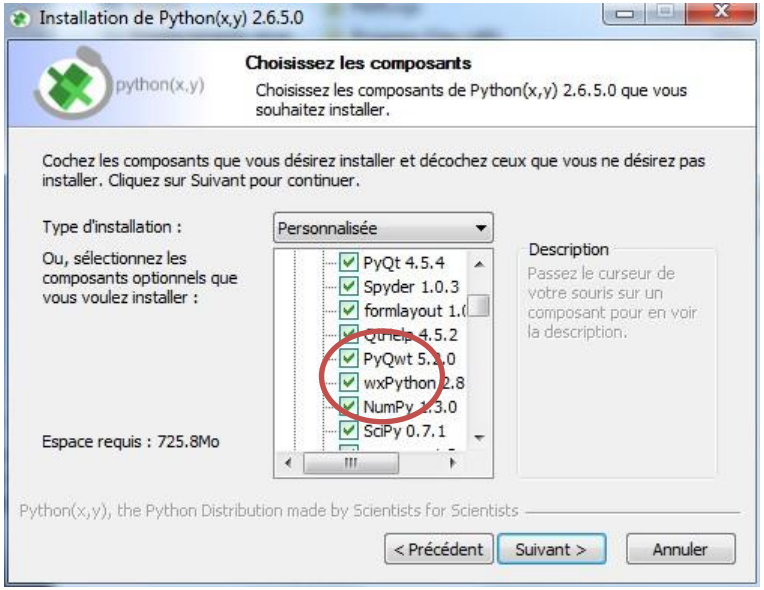
Pierre Raybaut (CEA / DAM)

Python(x,y) est une distribution Python à l'usage des scientifiques
Cinq fonctionnalités principales:

1. rassembler des bibliothèques Python et des environnements de développement complets adaptés à un usage scientifique
2. rassembler presque toute la documentation gratuite disponible sur ces librairies et outils ;
3. proposer un guide de démarrage en Python / Eclipse / Qt ;
4. configurer Eclipse pour qu'il soit prêt pour développer en Python, et modifier quelques paramètres Windows (tels que les associations de fichier, l'intégration dans l'explorateur Windows, etc.) ;
5. proposer un installeur tout-en-un, afin que l'utilisateur puisse installer ou désinstaller ces outils et fonctionnalités en un seul clic de souris.

Installation python xy





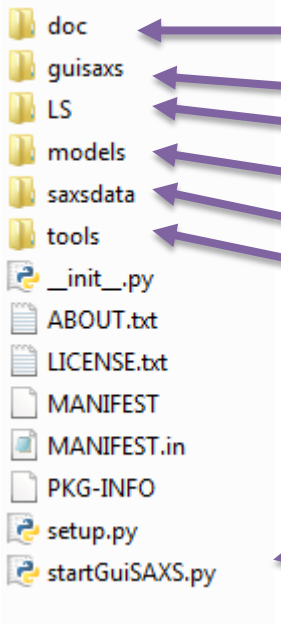
Installation pySAXS

- A partir du fichier zip (windows ou linux):
  - Décompresser le fichier zip
  - Lancer une fenêtre dos :
    - Dans windows 7 : executer , taper cmd
    - Aller dans le dossier de décompression
  - Taper `Python setup.py install`
- Installeur Windows

Le raccourci est dans C:\Python26\Lib\site-packages\pySAXS

startGUISAXS

Arborescence de pySAXS



Divers documents

Interface graphique

Cœur de pySAXS, notamment Lsca.py

Modèles

Données

Outils

Lanceur rapide de GuiSAXS

What is PySaxs ?

LS (LIONS Saxs) :

A special effort for a compilation of useful functions in Python

- calculating different form factors or structure factors :

```
def F1(q,R):
    """
    This function returns a scattering amplitude of a sphere of radius R for q
    """
    return (3.0*(numpy.sin(q*R)-q*R*numpy.cos(q*R)))/(q*R)**3.0
```

```
def P1(q,R):
    """
    This function returns the form factor of a sphere of radius R for q
    """
    if numpy.min(R)<=0.0:
        sys.stderr.write('can not compute for nul or negative sizes\n')
        return 1.0
    return F1(q,R)*F1(q,R)
```

Gives intensities (q range, parameters)

- For absolute intensities (scaling) processing

→ Can be used by researcher 'own' routines

PySaxs Models

Librairies of models:

- based on a Class model,
- using combination of Form factors and structures factors (LS)

What is a model ?

- I(q) function depending of parameters
- list of parameters with description, defaults values, name of authors
- Fitting functions based on scipy.optimize (simple with optimize.leastsq or with bounds : optimize.fmin_tnc)

Offering simple usage for fit :

```
from pySAXS.models import MonoSphere
sphere=MonoSphere()
y=array_of_experimentals_datas
res=sphere.fit(y)
```

How ?

1. A class model
2. And a mymodel.py file in the model directory

Warning :

- Computation time
- Specifics models

List of availables models (november 2011) :

- Capsule
- Core Shell Particle
- Core-shell cylinder
- Cube
- Cylinder with six levels
- Gaussian
- Mono Cylinder
- Mono Ellipse
- Multi: Doublet of identical spheres
- Multi: Tetrahedra of identical spheres
- Multi: Triplet of identical spheres
- Parallelepiped
- Porod
- Porod with curvature correction
- Spheres Monodisperse
- Spheres poly-Gauss analytique

Models

```python
class MonoSphere(Model):
    '''

    class monoSphere from LSSca
    by OT 10/06/2009
    '''
    def MonoSphereFunction(self,q,par):
        """
        q array of q (A-1)
        par[0] radius of the sphere (A)
        par[1] scattering length density of sphere (cm-2)
        par[2] scattering length density of outside (cm-2)
        par[3] concentration of sphere (cm-3)
        """
        if len(par)!=4:
            sys.stderr.write("This function requires a list of 4 parameters")
            return -1.
        else:
            return par[3]*(par[1]-par[2])**2.*getV(par[0])*getV(par[0])*1e-48*P1(q,par[0])
            #sys.stderr.write(str(par[0]))
            #return P1(q,par[0])

    '''
    parameters definition
    Model(0,MonoSphere,Qlogspace(1e-4,1.,500.),([250.0,2e11,1e10,1.5e15]),
    ("radius (A)","scattering length density of sphere (cm-2)","scattering length density c
    ,("%f","%1.3e","%1.3e","%1.3e"),(True,True,False,False)),
    from LSsca
    '''
    IntensityFunc=MonoSphereFunction #function
    N=0
    q=Qlogspace(1e-4,1.,500.)      #q range(x scale)
    Arg=[250.0,2e11,1e10,1.5e15]           #list of defaults parameters
    Format=["%f","%1.3e","%1.3e","%1.3e"]      #list of c format
    istofit=[True,True,False,False]    #list of boolean for fitting
    name="Spheres Monodisperse"          #name of the model
    Doc=["radius (A)",\
         "scattering length density of sphere (cm-2)",\
         "scattering length density of outside (cm-2)",\
         "number concentration (cm-3)"] #list of description for parameters
```

PySAXS Graphic User interface

A graphic user interface : GuiSAXS

no satisfaisant interface for data treatment and data manipulation

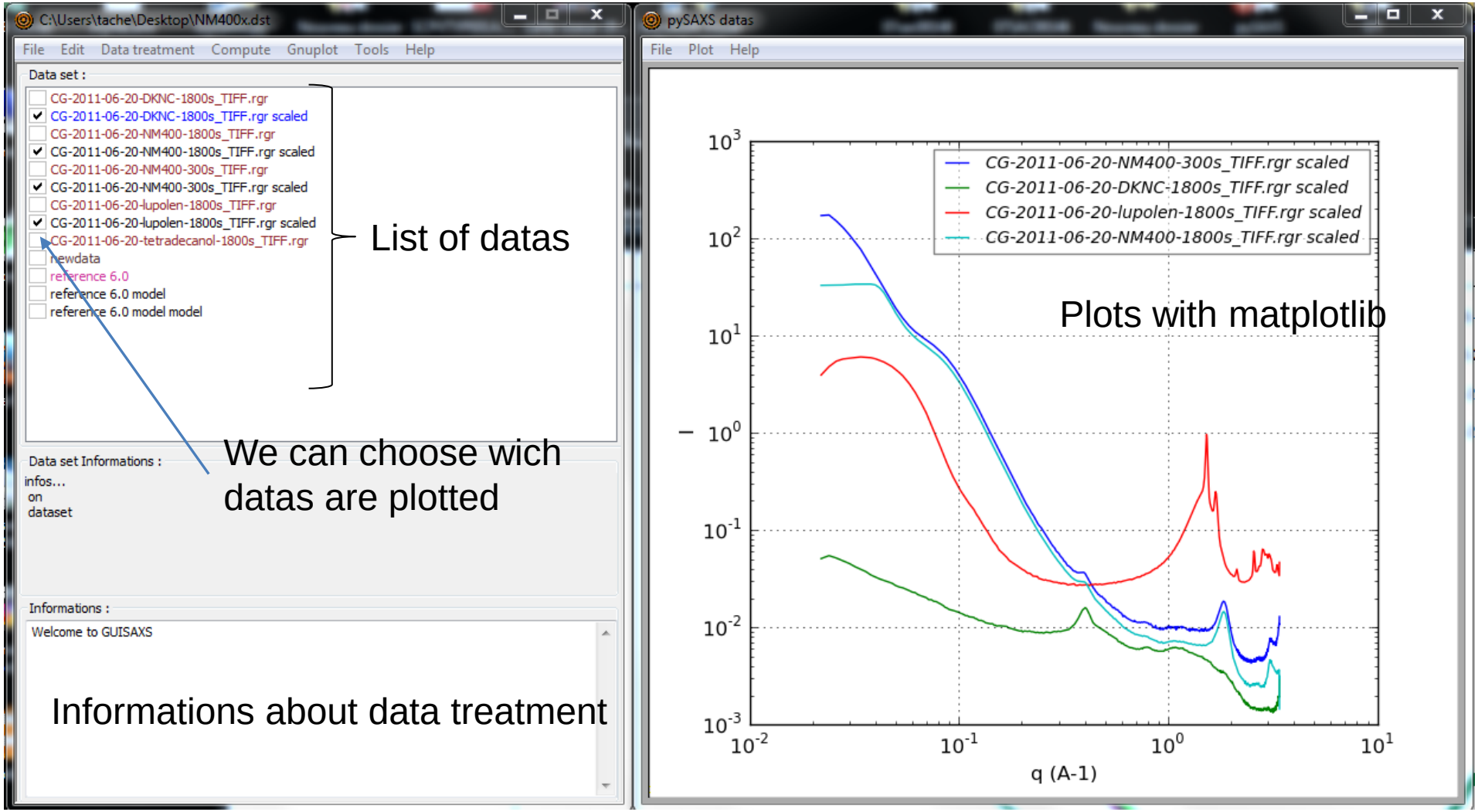
- import data from experiments (text file)
- scaling
- compare
- substraction or manipulation datas with different scales
- correct plotting tool (log scale)
- → gnuplot and matplotlib
- modeling

- informations about data treatment

Using wxPython :

Not a real choice
Works on windows and linux
No IDE : all the code is made by « hand »
→as much as possible : generic dialog boxes

GuiSAXS


C:\Users\tache\Desktop\NM400x.dst
File Edit Data treatment Compute Gnuplot Tools Help
Data set :
CG-2011-06-20-DKNC-1800s_TIFF.rgr
CG-2011-06-20-DKNC-1800s_TIFF.rgr scaled
CG-2011-06-20-NM400-1800s_TIFF.rgr
CG-2011-06-20-NM400-1800s_TIFF.rgr scaled
CG-2011-06-20-NM400-300s_TIFF.rgr
CG-2011-06-20-NM400-300s_TIFF.rgr scaled
CG-2011-06-20-lupolen-1800s_TIFF.rgr
CG-2011-06-20-lupolen-1800s_TIFF.rgr scaled
CG-2011-06-20-tetradecanol-1800s_TIFF.rgr
newdata
reference 6.0
reference 6.0 model
reference 6.0 model model
List of datas
Data set Informations :
infos...
on
dataset
We can choose wich datas are plotted
Informations :
Welcome to GUISAXS
Informations about data treatment
pySAXS datas
File Plot Help
CG-2011-06-20-NM400-300s_TIFF.rgr scaled
CG-2011-06-20-DKNC-1800s_TIFF.rgr scaled
CG-2011-06-20-lupolen-1800s_TIFF.rgr scaled
CG-2011-06-20-NM400-1800s_TIFF.rgr scaled
Plots with matplotlib
I
q (A-1)

GuiSAXS

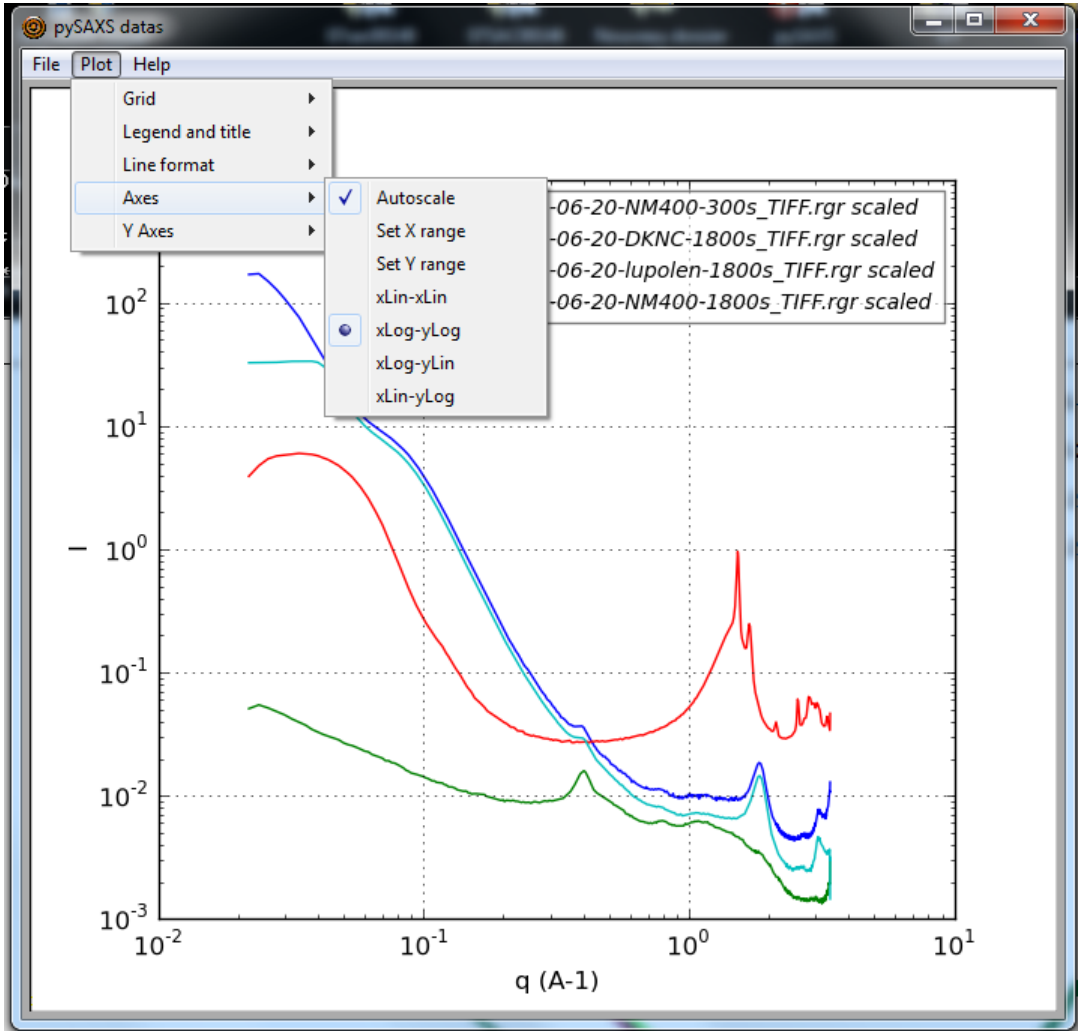
A matplotlib frame with a menu where you can :

- Add a grid
- Change legend and title
- Set the line format
- Set the axes scales
- Save as picture

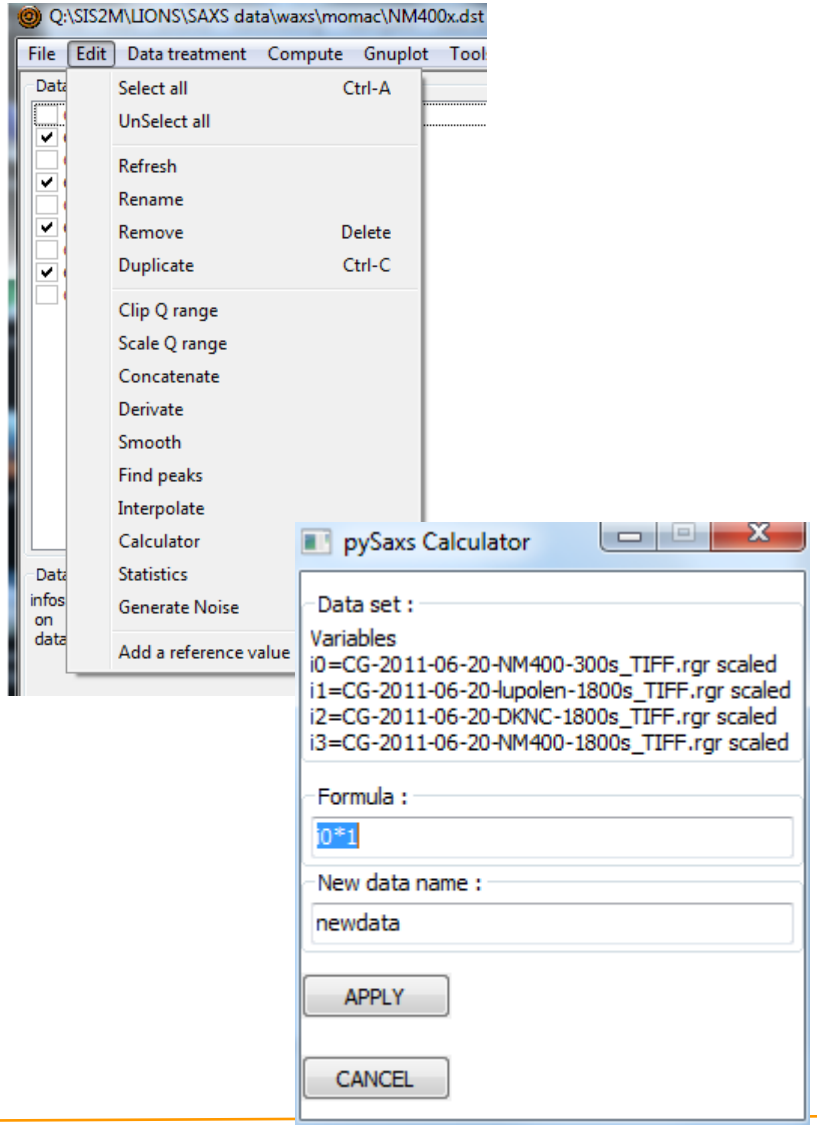
- Colors are automatics,

it is not possible to change them
Can be improved

OR you can use gnuplot windows



GuiSAXS : data manipulation



- Refresh : reload datas from file
- Rename
- Remove datas from list
- Duplicate

- Clip q range
- Scale : change scale (q*10)
- Concatenate
- Derivate
- Smooth datas
- Find peaks
- Interpolate (add points)
- Calculator :
  open a dialog box and let the user specify a formula for data manipulations
- Statistics
- Generate noise on datas
- Add a reference value : to compare with a flat datas

GuiSAXS : data scaling

$$I(q) = \frac{C_{ij}}{\phi_0 . dt} . \frac{1}{\Delta\Omega} . \frac{1}{e}$$

- C_{ij} is the number of counts detected on pixel ij during dt <u>with background substracted</u>
- ϕ_0 is the transmitted flux (photons/s) by the sample

  $\phi_0 = \phi_{incident} . T.K$

  T is the transmission of the sample

  K is the detector quantum efficiency

  $K = \frac{\eta_1}{\eta_2}$

  η_1, is the detector quantum efficiency for the counts C_{ij}

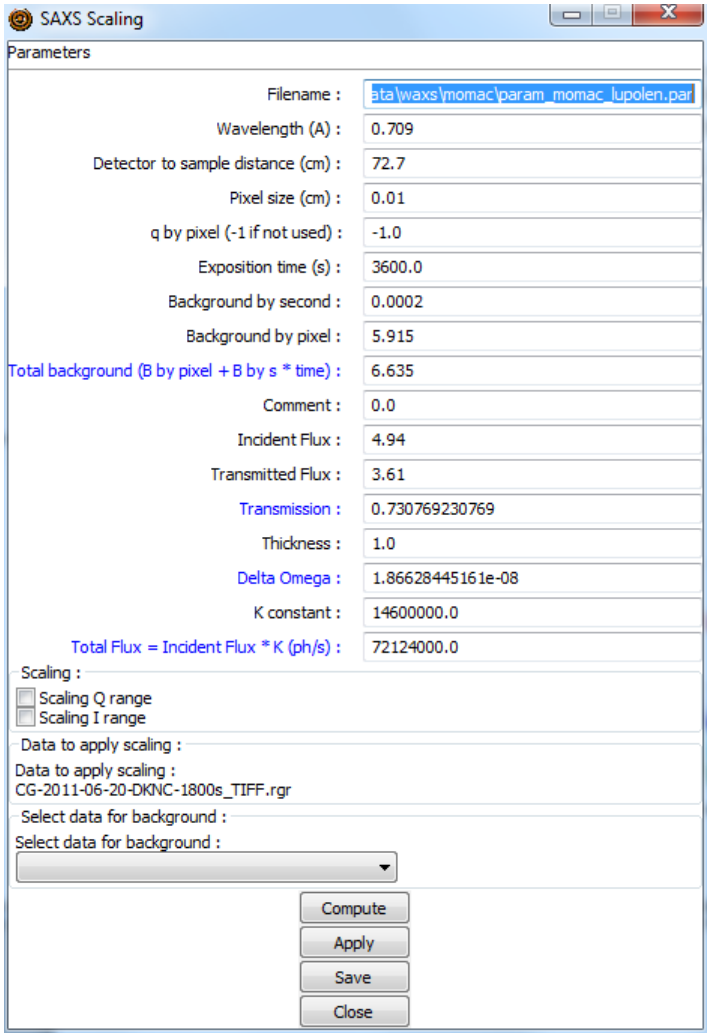
  η_2, is the detector quantum efficiency when measuring the incident beam.
- $\Delta\Omega$ is the solid angle covered by one pixel seen from the center of the sample.

  $\Delta\Omega = \frac{p^2}{D^2}$

  p is the pixel size and D the sample to detector distance
- e is the thickness of the sample (cm)

```
Intensity=(n-background)/(time * DeltaOmega * Transmission *
Thickness * Flux * K)
```

→ Absolute intensities are independant from experiment



Data substraction

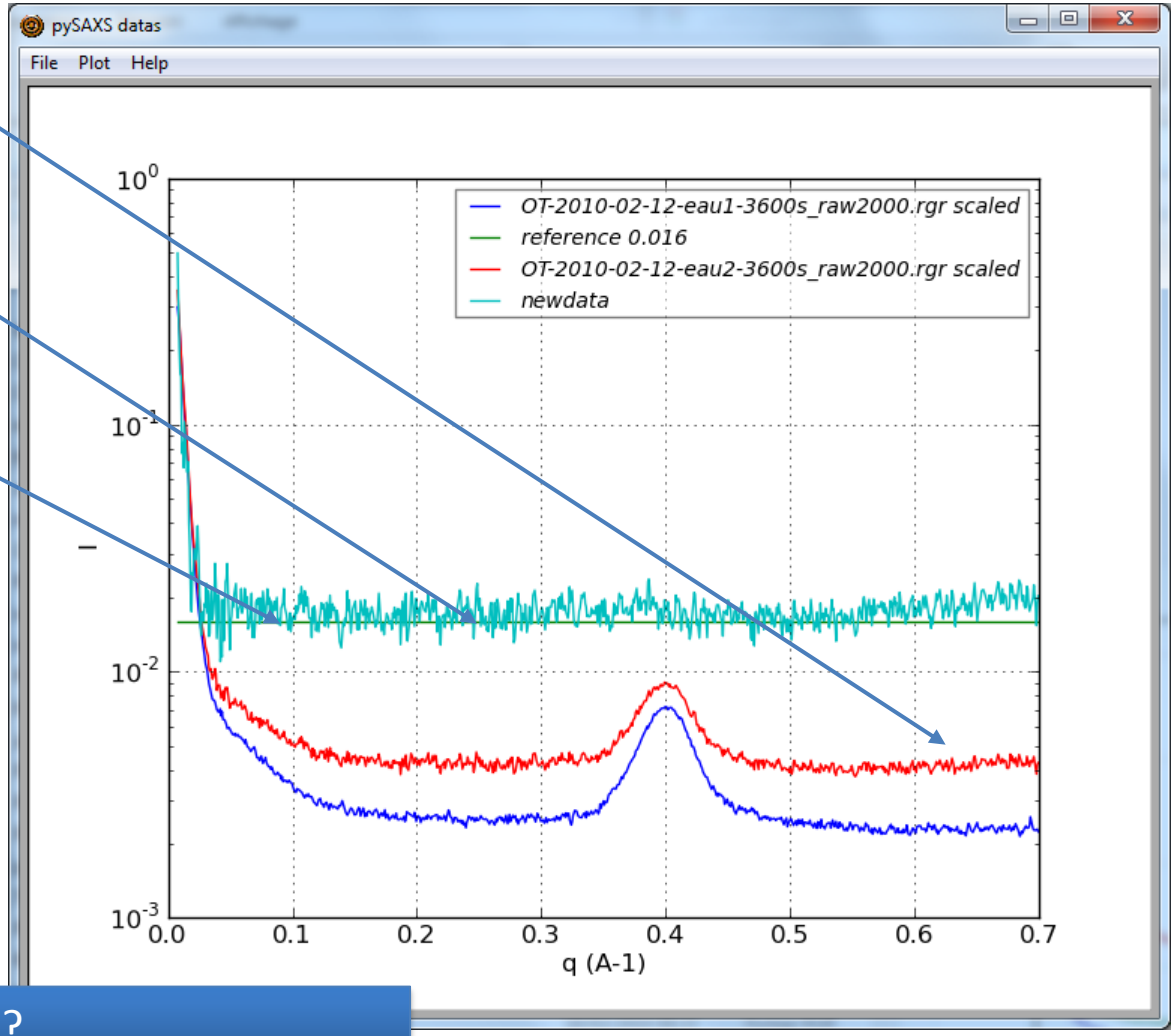
Data for 1mm and 2 mm thickness of water

Substraction (gives 1mm of water)

Reference (calculated value)

Data processing is done by using interpolation of datas

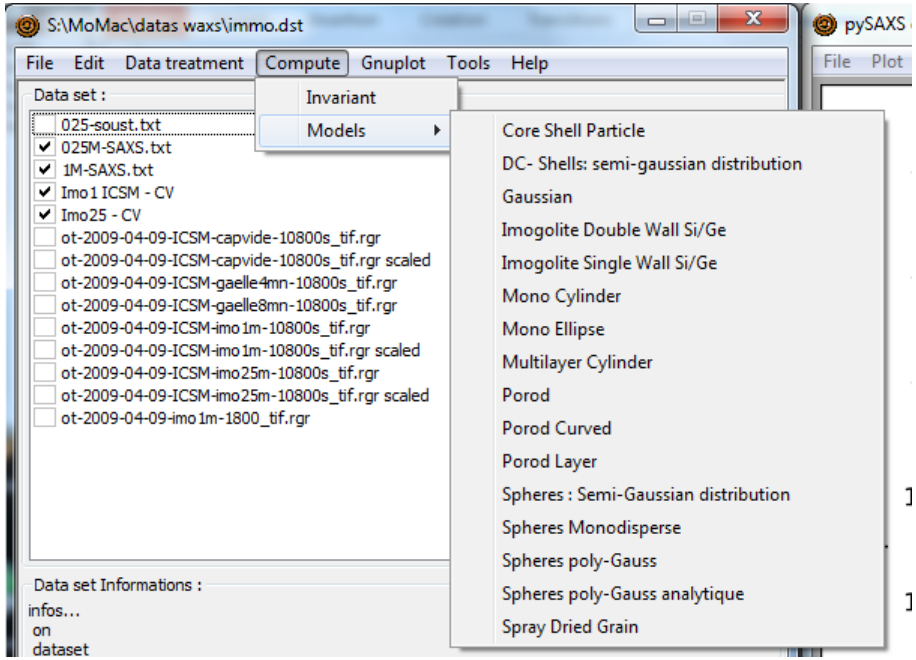
With propagation of measurement's errors



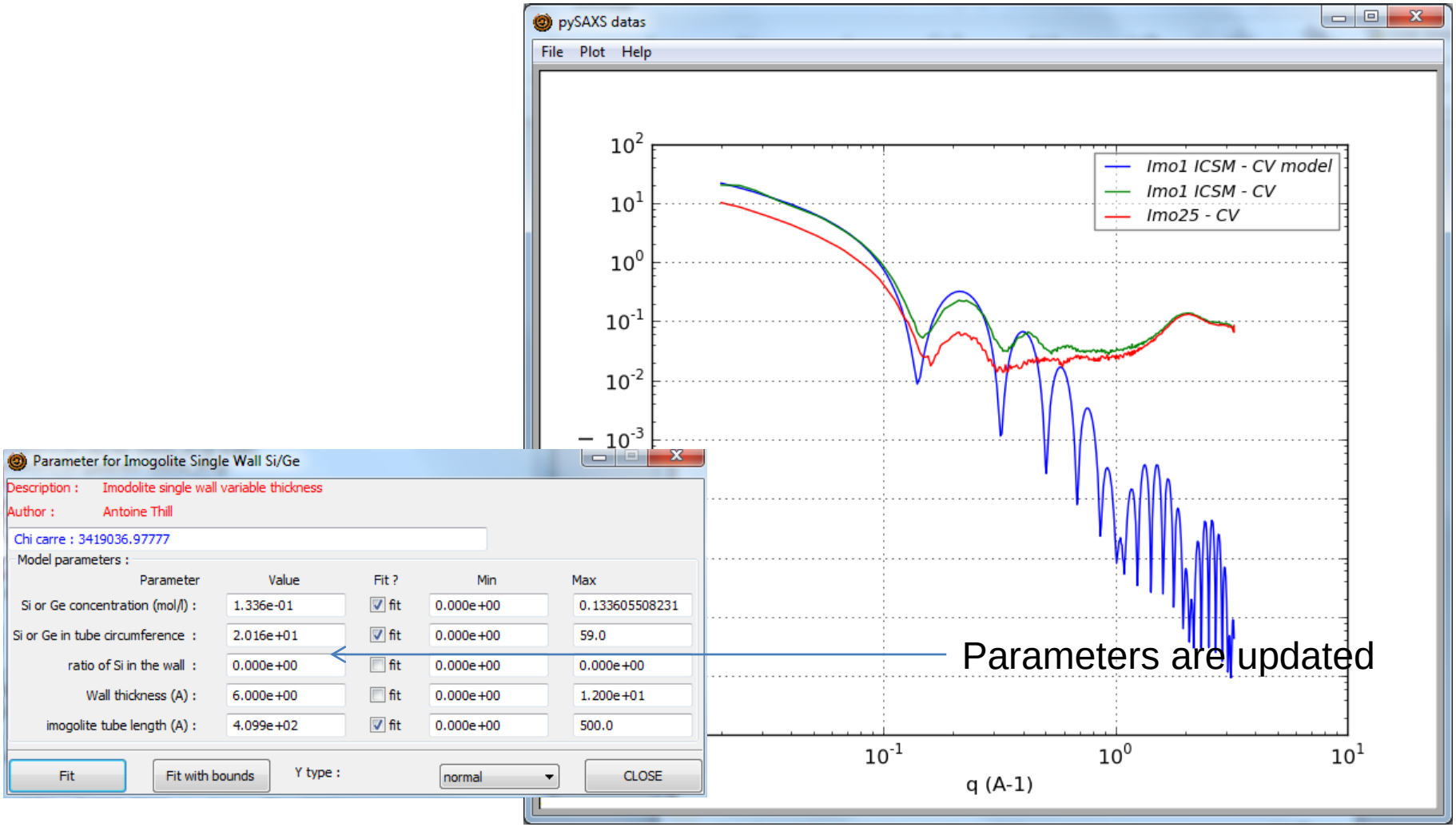
→ How could we do that in excel ?

Fitting with models

Models integrated automatically in the menu



Fitting with models

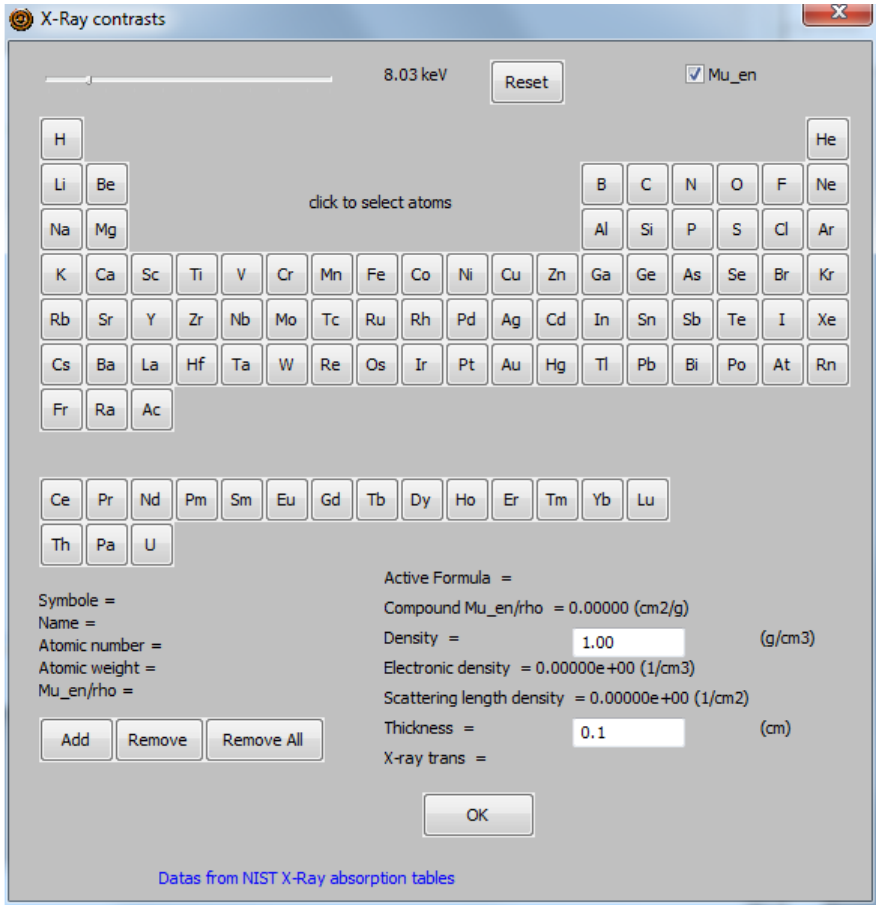


Other functionalities :

- Datas saved as txt
- Datas saved by group (dataset) in xml file
- **Keeping measurements error bar**

- X-Ray contrasts dialog box : for calcutating transmission of sample depending on composition and x-ray energy

- PySAXS is given to users
- Easy Installation on Windows with PythonXY



Improvements

- Basic models
- Identify specific models

- Xml format for dataset (list of datas) :
  - Keep definition of parameters
  - Linux and windows compatible
  - Linux distribution