SAXS USER BASICS

X-Ray generator is already ON (photo + 

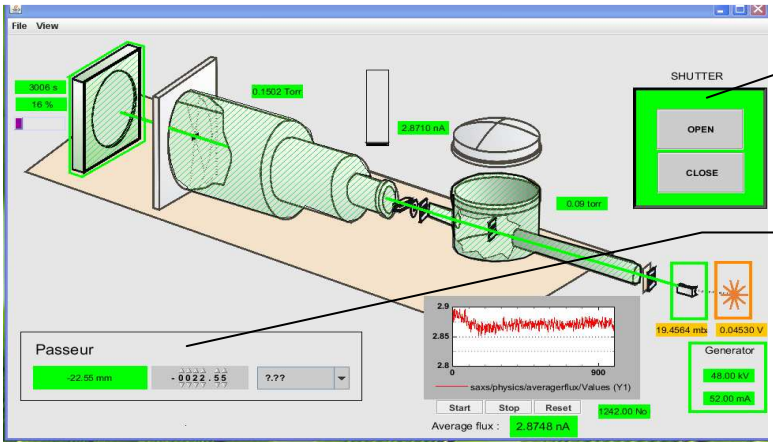
)

Computers are ON and softwares open (python  , experiment.py)
Control box on External

Computers

- SAXS computer (left) : to conduct SAXS experiments

Python shell (Idle) = commands for acquisition with python
Synoptic = where the current values (transmitted flux, generator state, time left for experiment, etc) are displayed :



Command to open/close the shutter

Command to move the sample holder to a defined position

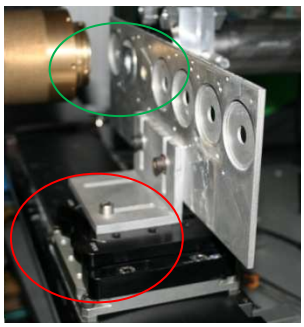
- MAR computer (right) : dedicated to the detector

ATK panel displays the last image scanned. Raw images are saved in the computer (***data/mar/images***)
Green = ready; blue = scanning or erasing; yellow= not ready (need to reset)

- TANGO computer (back of the room): to treat the data with *ImageJ* and *GUISAXS*

Setting up samples

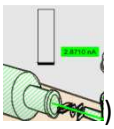
Lock/unlock carefully the sample holder (set position 120 mm = base closest to the user), screw/unscrew the vacuum "nose" to minimize the air thickness between sample and vacuum chamber.



Position 120 mm

Running an experiment

- Place the sample holder containing your samples and/or references, close the door.
- Open the shutter 

(control the flux on the photodiode ).
- If necessary use ScanPosi to find the exact position for “empty”, references and/or your sample :
    - In Python Shell, write and return to run :

```
From experiment import *
scanPosi(type here the min position, type here the max position, type here the step)
```

- Open a previous script (Python Shell ***file/open/recent file***) and save it under your initials (then overwrite for each modification).

script_MODELE.py

```
from experiment import *

#measure transmission
measureTempty(68,200)

#erase mar
saxs.mar.erase()

#start
count("samplename",3600,position=71.39,init='CG')
#count("Lupolen",1800,position=pos5,init='CG')

#measure transmission
measureTempty(68,200)

#erase mar
saxs.mar.erase()

#saxs.mar.off()
#stopGeneX()
#end
```

Fill in empty position and counting time (s)

Fill in sample name, counting time (s), position (mm), user initials ➔ generate the filename CG-2010-03-19-samplename-3600s

If necessary add command lines for other references or samples

Fill in empty position and counting time (s)

#count("Lupolen",1800,position=pos5,init='CG') ➔ red= comment not read by TANGO

- ***Run/run module*** *to run the experiment*

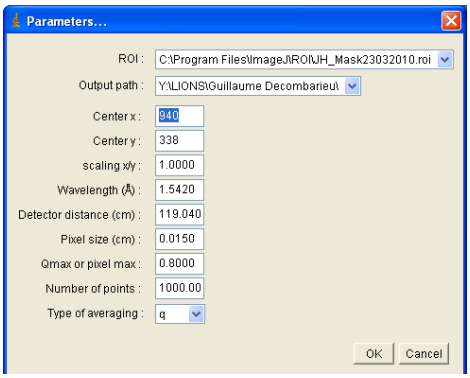
CRTL C to stop the running module

- **Write the experiment parameters for each sample in the SAXS notebook**, together with the incident flux (“empty”) and the averaged transmitted flux displayed in the Python Shell window.

Treating the data

- On MAR computer : copy-paste image file from ***data/mar/images*** to ***laborx/communs/SIS2M/lions/saxsdata/saxs/mar*** (password needed)
- On TANGO computer : Open ImageJ , ***file/open*** the image file
- ***Image/look up tables/royal*** *and* ***Image/Adjust/Color Balance*** to adjust the image

- **<u>For the first reference (calibration in q-range) :</u>**
  - Adjust the center of the image (around x= 940, y= 338)
  - Define a mask , in ROI manager, open the most recent mask in ***program file/ImageJ/ROI, c***heck the limits of the mask. To modify : right click/ ***split***, delete previous one, use to move the points, ***update*** , then right click/***combine***, create , then save under a new name.
  - Radial average on q :



Check mask file and output path to save the .rgr file

Detector-sample distance calibrated with the reference for q-range (octadecanol or tetradecanol)

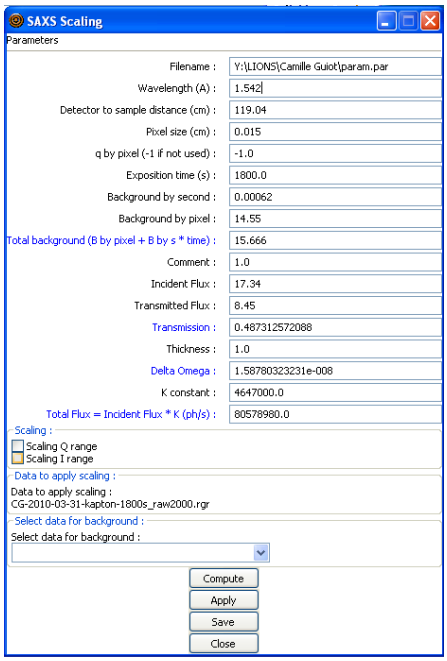
OK **creates a .rgr file**

- **<u>For all the following images :</u>**

Just open the image and radial average on q (the parameters and the mask have been defined already).

With GUISAXS (scalling and normalization)

- open your .rgr file(s)
- select the data to be scaled, ***Data treatment/SAXS/scaling,*** open a previous set of parameters:



*Fill in **exposition time, incident and transmitted flux** and check that the other parameters are correct, then click compute to automatically calculate internal backaround and transmission*

K constant is defined by the intensity calibration with lupolen

Tick Scalling I range *and Apply to generate the scaled curves*

- Calibration : check the q_{max} position for the references in q-range (octadecanol = 0 ,1525 $Å^{-1}$ tetradecanol = 0 ,1583 $Å^{-1}$), and the maximum intensity for the reference in intensity (Lupolen, 6 cm^{-1}) (q-range is calibrated through the sample-detector distance and intensity is calibrated by K constant).
- perform operations on curves (background subtractions, thickness normalization, etc) ***Edit/calculator***
- Save a .txt file for the ticked data (***file/save***), that can be later treated with excel, or save the set of data (***file/save all***) to open them later with pysaxs.