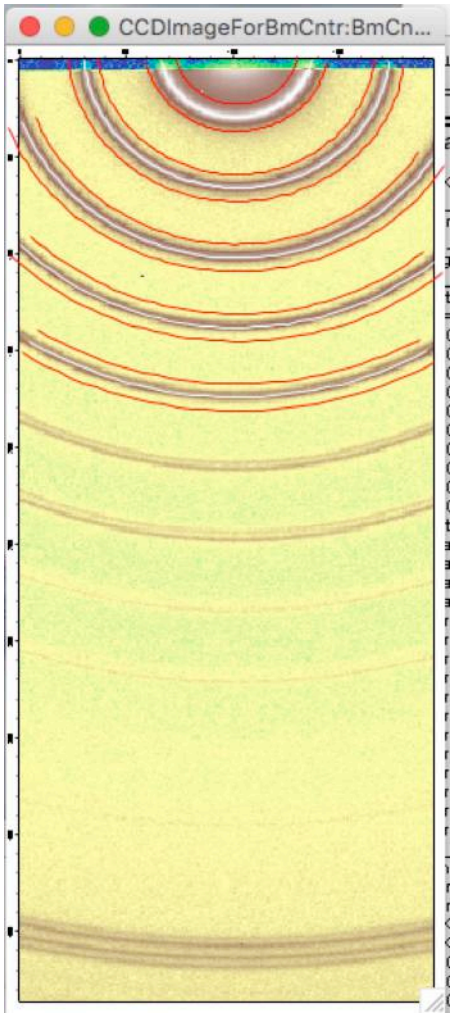


“Nika” manual

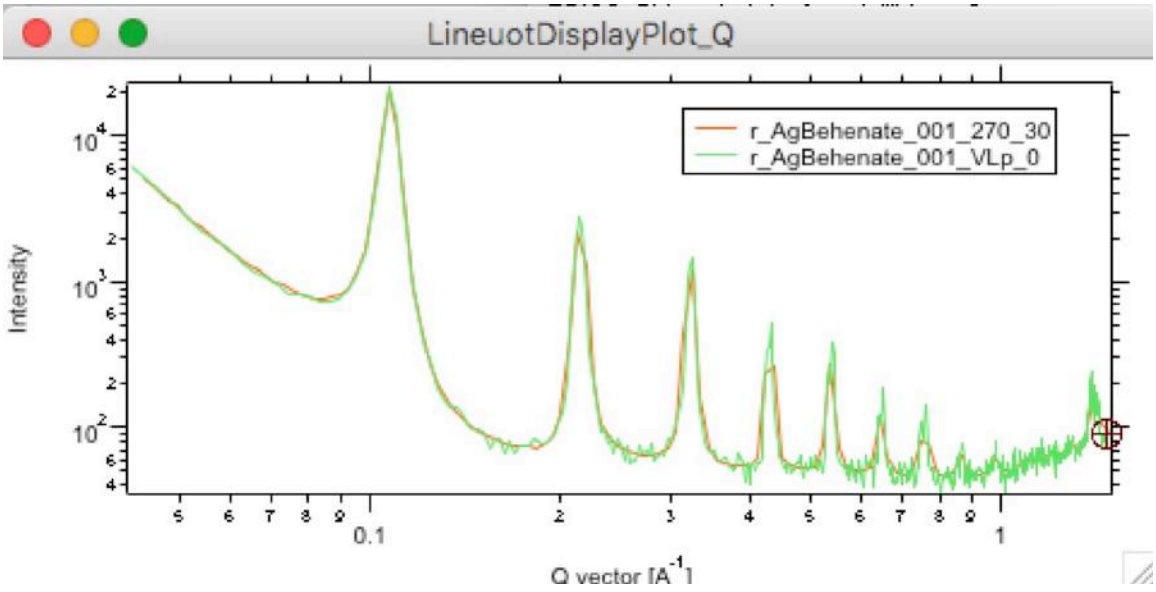
Version 1.70

Igor 6.37 and higher compatible

Date: Sunday, October 25, 2015

Jan Ilavsky

Ilavsky@aps.anl.gov



This is manual for set of macros developed for Igor Pro (Wavemetrics, Inc, www.wavemetrics.com) version 6.30 and higher. These macros are designed to process 2D (CCD and other area detectors) data from small-angle and wide-angle scattering instruments. The purpose is to process the 2D data and convert these into 1D "line outs" data – providing correctly calibrated Intensity, q (two theta or d), and errors.

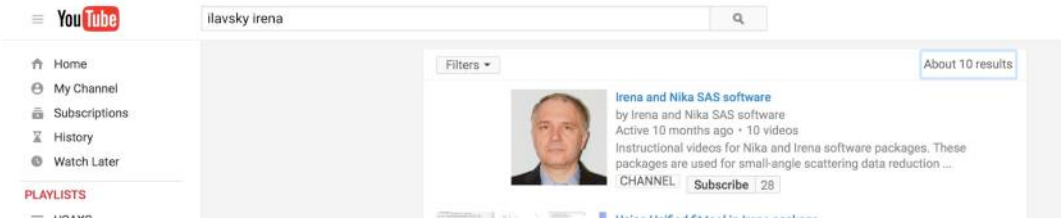
Nika was designed to provide number of methods to extract the data:

1. Method of Sector and circular averages
2. Intensity along linear and elliptical path (vertical/horizontal lines, line under and angle and ellipse of arbitrary aspect ratio)
3. Intensity along linear path but for Grazing incidence geometry
4. Intensity vs azimuthal angle image intended for manual inspection of geometry.

Content:

Instructions available on Youtube channel

I have Youtube channel for instructional movies. Search for example “Ilavsky Irena” on Youtube and you will see something like this:



The link to the channel is here:

https://www.youtube.com/channel/UCDTzjGr3mAbRi3O4DJG7xHA

for those who can click thin in pdf file…

This channel contains instructional movies how to install the package and how to use different tools. Please, watch it if you need help. It may help you

Courses available on various places in the World

Over the last few years I have had many courses at the APS and around the world either at institutions or at conferences. These, typically two day courses, teach how to use Irena. Some news about these courses should be available on:

http://small-angle.aps.anl.gov

http://small-angle.aps.anl.gov/courses/Irena_Nika_courses_report.html

Installing macros

To install the macros, please install first Igor Pro, version 6 and update to latest release (6.37 at this time).

There are few ways to install the macros:

Easy way using Igor 6.30 and higher (including user with limited privileges)
Download the “Universal installer.pxp” file from my web site. Optionally, if you have firewall issues or just want to have easier life, download ALSO the whole depository of my packages as one zip file. Then use this Igor experiment and install using the buttons provided. Note, that if you have local copy (unzip the downloaded (large) zip file, then select “Use local copy” checkbox. Instructions are in the file itself. It usually works just fine, but sometimes ftp communication is either bad due to network issues or even prevented due to firewall, so then local copy is your only choice. Install the xop support, please…

II. May be it may work … using Java instaler.
Download Java installer from my web site and use the installer. Install packages you need AND xop support.

III. The hard way, when the other methods fail... Zip files.
Get zip file for Irena package AND xops, appropriate for your platform. Place the files in the zip file, following the folders in the appropriate places in the Igor Pro Folder in User area. This location is easiest found by using in Igor Pro in help menu the item "Show Igor Pro User Files". Note that some of the files belong to Igor Procedures and some in user procedures, keep folder structure as is in the zip file, please...

NOTE: If you had prior installation (before 6.10 version of Igor) : Update Igor Pro (free from any 6.xx version) to latest version and check for presence of obsolete version :
Locate Igor Pro Files (again: Help menu in Igor, Show Igor Files) and remove any files related to Irena, Nika, and Indra from Igor Procedures and from User Procedures. This should be done automatically by the installers, but may not be possible if you are running as lower privilege user installers may not be able to do this.

To load macros, **select “Load Nika 2D SAS macros” from “Macros” menu** after starting Igor Pro. Whichever method you choose, the macros should work the same.

Please, learn more about full capabilities of the Igor Pro. It is very powerful graphing and data evaluation package. It may be necessary for you to handle data import and handling, data export and some graphing. Further, the macros heavily rely on the data folder structure, so it is important to learn enough to realize the use of this feature…

Optional xop for MarIP (image plate) format
Mar Image plate format is really cumbersome and cannot be efficiently read by Igor code, therefore I have initiated development of xop to read this image file. This xop was

developed by Jonathan Guyer and the code is available from : http://www.igorexchange.com/project/CCP4XOP . It is, however, due to licensing of one library distributed as source code and needs to be compiled by the user. If you do not want to compile the code your self, send me an e-mail request and I will send you compiled xop for your platform. I believe this complies with the licensing of this ccp4 library.

This xop belongs to folder :
…../Igor Pro Folder/Igor Extensions

Comment: version 1.42 (10/27/2009) – I was unable to recompile this xop for the latest version of OSX and long file names again. Therefore this xop is likely not available for Mac at this time.

Loading the macros

Install macros (see above)
Start Igor Pro
In menu “Macros” select “Load Nika 2D SAS macros”.
New menu “SAS 2D” appears. This is where all the Nika macros are controlled from.

Removing the macros

There are two items on may wish to do to unload the macros from any experiment. First to remove the large lookup tables associated with the methods Nika is using to convert 2D to 1D data. This is done by selecting “HouseKeeping” from “SAS 2D” menu. This will make the Igor experiment much smaller (often by 60 Mb or even more).

Then removing the macros itself is achieved by selecting “Remove Nika 1 macros” in the “SAS 2D” menu. This will unload macros and put back in the “Macros” menu command to load Nika macros, if necessary.

Configure default fonts and names & Errors

“**GUI & uncertainty config**” in the SAS menu will create panel with some controls common for all tools, like font type & size and how legend names are handled. NOTE: Panel controls are applied immediately to all existing panels, graph controls are applied ONLY to the newly created graphs (and only those which were upgraded to this behavior).

Panels font and font sizes

These controls enable user to customize font used on control panels therefore this enables customization for a given platform. This is necessary as more and more control is provided on each platform to user and therefore default fonts and font sizes may not be appropriate any more for the panels I design. These settings are actually saved on a given machine as well as the experiment. This has some interesting features, so please, read carefully:

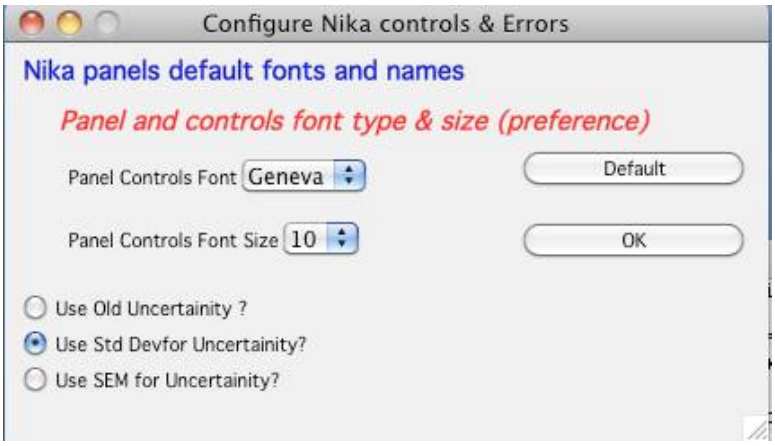
When these controls are run (and user is forced to run them if the Irena is loaded and preferences are not found), they save preferences in special folder Igor maintains for users. At the same time, the settings are applied to the current experiment.

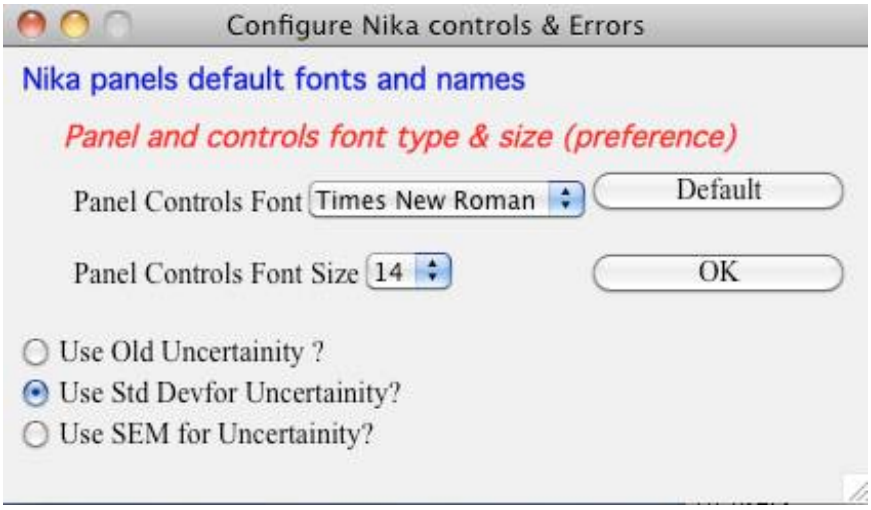
When this experiment is opened on another computer, the preferences from that computer are not reloaded, so the experiment will use preferences from the original computer. When the "Configure GUI and Graph defaults" is run, it will reload the computer defaults and apply them to the given experiment. Then user can change the fonts and font sizes as they wish. The new settings are saved on the computer – and within the experiment.

Note, that Panel font and font size are platform specific, so same experiment may present differently looking panels on Mac and PC.

Note, not all controls actually follow these settings, I have been changing some buttons to specific font and font size and those are not affected by these settings.

If there are any issues with the behavior, please, let me know and I'll see if I can make it more logical.





Note the difference in Configure GUI and Graph defaults panels when different fonts are used. Left is using Geneva font size 9, right is using Times New Roman size 14, both on Mac platform. You can mess up the panels really well by wrong choices!

Defaults button returns the panel font choices to platform specific default state (Mac: Geneva size 9 and PC Tahoma size 12). Note, that there is no guarantee that these were your choices before. But these should be reasonable choices for most setups.

Error (uncertainty) estimates for points

Up to version 1.42 Nika used "error" calculation, which seemed to produce reasonable numbers inmost cases. But one of the users realized, that it has error in it (bug) and fails in cases of really low intensities, which can be assumed to be Poisson distribution. Therefore, from version 1.43 Nika has three options for this calculation.

1. Old method (default for compatibility reasons). As I said, it has a bug, but seem to work most of the time well enough.
2. Standard deviation
3. Standard error of mean

Please note, that Standard error of mean (SEM) is miserably small for high-intensity instruments and generally is too small. At least in my experience. However, it seems to work fine for Pilatus detectors. Standard deviation was what I was trying to get with the old method – there should not be much difference between them most of the time.

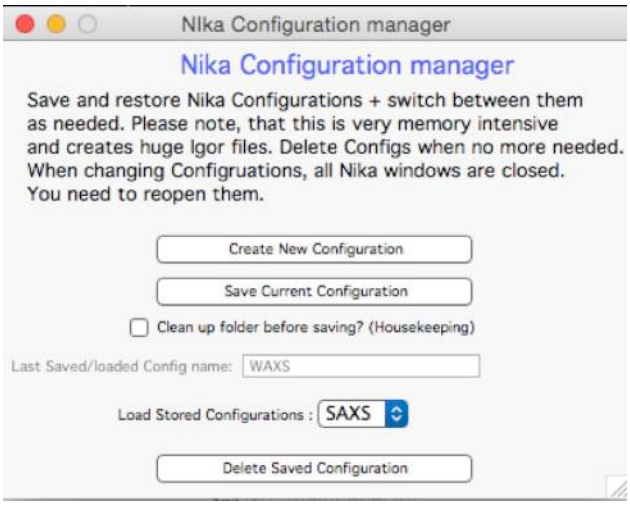
Configuration manager

From version 1.70 Nika contains "**Configuration manager**" . This tool enables one to configure multiple "versions" on Nika inside one igor experiment and switch among them. The main purpose is to handle cases when user needs to reduce data from multiple distances of the same instrument, multiple detectors on the same instrument, optionally from multiple instruments. We have such case on APS 9ID USAXS/SAXS/WAXS instrument where we need to reduce data from SAXS and WAXS.

How is this done?
To achieve this purpose the Configuration manager will copy whole Nika working folder – create snapshot - (root:Packages:Convert2Dto1D) into specific location (root:Packages:NikaGeometries) with user selected (cleaned up!) name. This contains everything in that folder and it is all Nika needs to operate. This therefore includes mask, lookup tables, Empty image, Dark image - everything… Of course, it is a snapshot of the status at that given time. User can then reconfigure Nika to have another distance, detector – whatever – and save this as second configuration. Then user can switch between the geometries as needed.
NOTE:
Only one configuration can be run at given time.
All windows will be closed when switching.
The Igor files are likely to get very large (this is space expensive process).

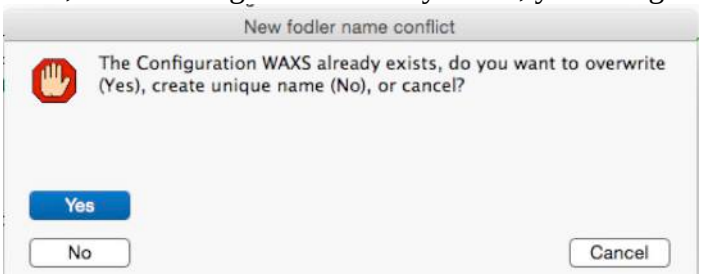
The save is done at a specific moment in time, so later changes in configuration are not reflected and user needs to save the configuration again.

Here is the GUI:



"**Create New Configuration**" button will create new default Nika "Configuration" – it will DELETE the current one and restart Nika. Use this button carefully. But it will guarantee that user starts with "clean slate".

"**Save Current Configuration**" button will save the currently configured Nika folder into the location explained above and make it available as "Stored configuration" User can give it their own name – it will be cleaned up and if necessary (with dialog) also unique.
Note, if the Configuration already exists, you will get dialog:



Choosing YES (= overwrite) will delete the existing Configuration of this name and save the existing one under that name. Choosing No will create unique name by adding order number at the end (from WAXS will create WAXS0, WAXS1, etc…). bCancel will cancel of course.

"**Clean up folder before saving**?" checkbox – This will run the "housekeeping" function before storing the Current configuration. This will remove the temporary lookup tables and other stuff, which can be easily recalculated when needed (at cost of CPU). This may make the Igor files much smaller - but when restored the first image processing will take longer.

"**Last Saved/Loaded Config name**" this shows what was the last name under which this Configuration was saved or loaded. This is bit dangerous name – it can become stale

really quickly as it has no idea what user has done and what therefore may have changed on the configuration. Imagine you have two detector distances. You create Nika configuration for one of them (Short distance) and save it. That will show now “Short distance” as the name of current configuration… But then you proceed to change the distance (and other parameters). This string will not change – it has no idea you changed the parameters. So keep this in mind. It is simply how you called the last configuration when it was saved or loaded.

“**Load Stored Configuration**” – popup which lists saved Configurations. If you select configuration from this popup, dialog will come up which will enable you to : 1. Cancel if you do not want to do anything, 2. Save current configuration and load the one you selected, 3. Not save the current configuration and load the selected one. In any case, you can cancel before anything is done.
When loading saved configuration all Nika windows should be closed and after restoring the configuration, the main Nika panel reopens.

“**Delete Saved Configuration**” This will open dialog where you can select saved configuration and delete it. This way user can remove stored configurations when they are no more needed to save space (or confusion) in Igor experiment (file).

There is no way at this time to rename saved configurations, except manually renaming the folder in the location (root:Packages:NikaGeometries). The Configuration manager needs to be restarted after this to figure out the new names.

Modifying Nika functionality

Basic method of modifying Nika functionality is to use "hook" functions – functions, which are called when they exist. In order to use them one needs to be proficient Igor programmer, so the description here is limited. You will need to read the code anyway. Note: if you need more "hook" functions, ask and I'll add them. Better then forking your code and then not being able to update your code in the future.

List of hook functions:

Name of hook function	Called where	Why?
Nika_Hook_ModifyMainPanel()	NI1A_Convert2Dto1DPanelFnct()	After the main panel is created, so user can change the panel as needed.
Nika_Hook_AfterDisplayLineout(int,Qvec,Err)	NI1A_DisplayLineoutAfterProc	After lineout is displayed so user can change it as needed
ModifyImportedImageHook(ImageName)	NI1BC_BmCntrCreateImage NI1A_ImportThisOneFile NI1A_LoadEmptyOrDark NI1M_MaskCreateImage NI1_FloodCreateAppendImage	Modifies image after import. E.g., enables user to trim image to ROI only etc.
PilatusHookFunction(ImageName)	NI1A_UniversalLoader	After Piltus image is loaded so one can modify it as needed.
ImportedImageHookFunction(ImageName)	NI1A_UniversalLoader	After loading any image just at the end. Can be used to modify loaded image as needed.
AfterDisplayImageHook()	Various places after Nika displays detector image.	Can be used to modify displayed image. Note – works on top image.
Movie_UserHookFunction()	NI1A_MovieCallUserHookFunction	In movie tool – create or modify image used for movie. See GUI.
Need more?	Let me know here…	

Example:

Following function is called after any image is loaded (if it exists) and simply prints in history area image statistics.

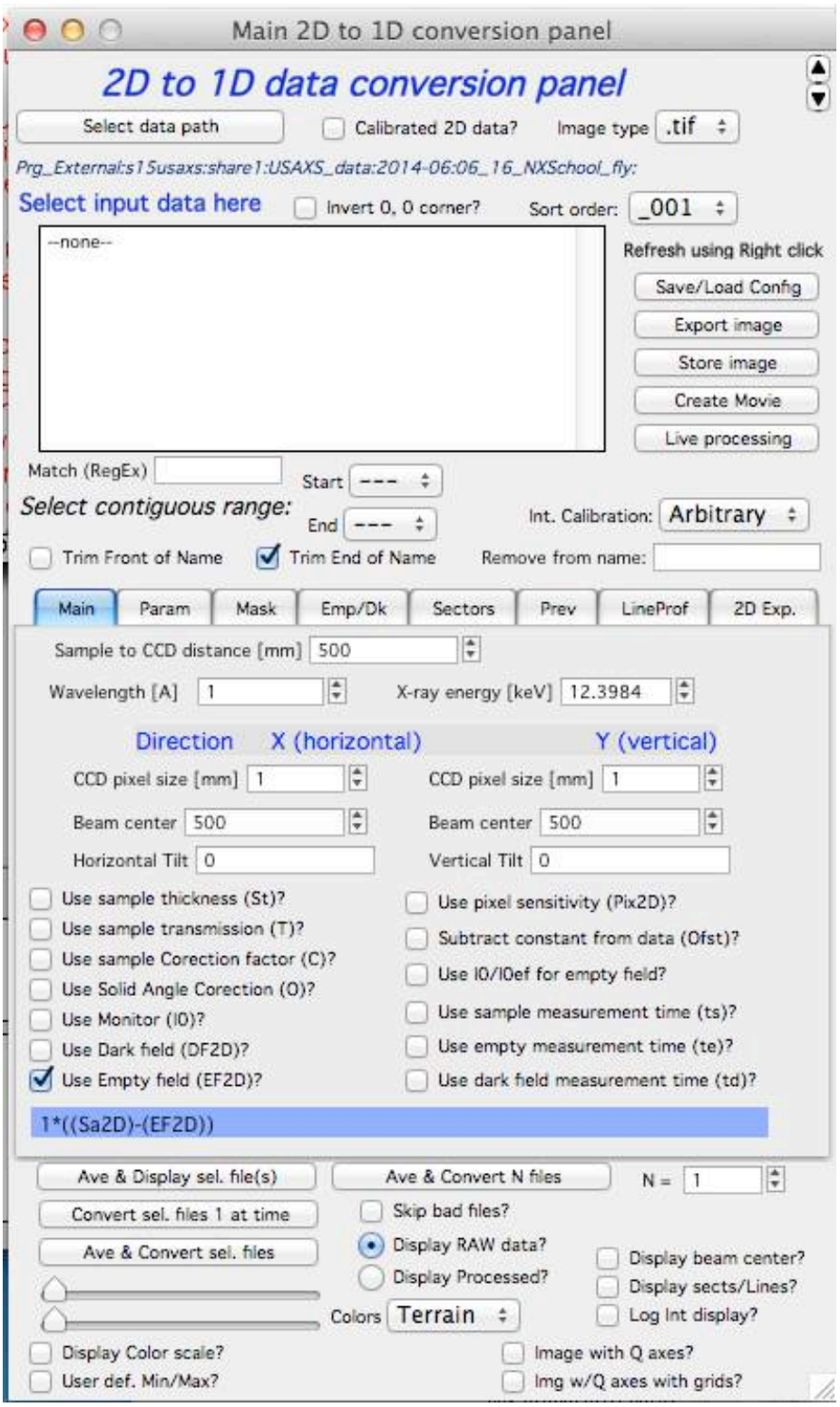
```
Function ImportedImageHookFunction(NewWaveName)
        wave NewWaveName

        wavestats  NewWaveName
end
```

Following function, if present, will zoom in top 50 pixels on the detector image

```
Function AfterDisplayImageHook()
        SetAxis/R left 50,0
end
```

Main Nika Panel
Select “Main panel” from the “SAS 2D” menu. This will present the following panel:



The panel has three major parts:

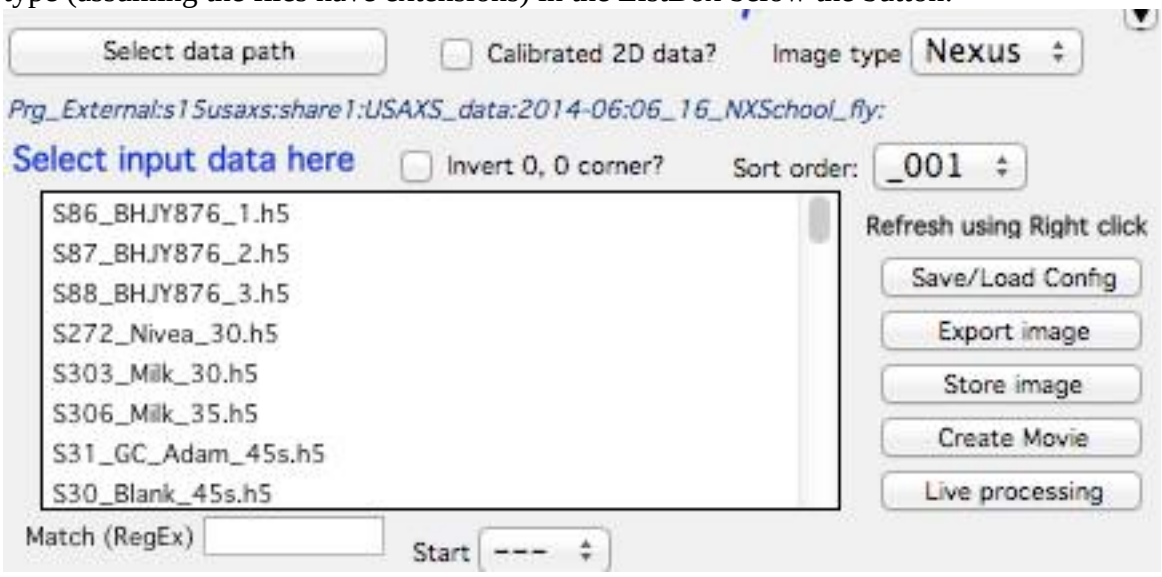
1. Top is designed for 2D data selection. Here user selects which 2D image will be processed.
2. Middle (tabbed area) is designed for controls of processing. This is the busiest area of the panel and each tab will be explained later.
3. Bottom contains buttons for main controls and 2D image controls.

Selecting data

The macros can load selected data types. If appropriate file type is not found in the "Image type" popup menu at the top right corner, you will have to contact me so I add appropriate loader for your data.
Select appropriate type of data you have and then push "Select data path" button, dialog is presented, in which path to folder on the hard drive containing 2D images is selected:

NOTE the "***Calibrated 2D data***?" checkbox. If selected, Niak expects 2D calibrated data – fully normalized and corrected data provided as one of the 2D formats, basically 2D image of Intensity, Q (vector), and uncertainity. Number of options is being current developed, the code currently handles EQSAXS (ORNL) and canSAS/Nexus. This part is under heavy development at this time, expect changes…

When valid path is selected, the Igor will check the folder and list all files of appropriate type (assuming the files have extensions) in the ListBox below the button:



Here user can select one files, more files (by holding down shift key on Windows) and continuing selection (using the two pull down menus below the list Box)…

Note, that from Nika ver. 1.66 Listoboxes have right click actions and users can refresh content and perform some functions from right click.

Use the “Match” field to mask the file names with Regular expression. To match part of the name, just use the string needed - so matching samples with _15s in name, just add _15s in the field. Regular expressions are very powerful, read on line how to use them.

Note, the files ending with “_mask”. These are mask files created by Nika package, these were used to be tiff files, now they are hdf5 files… Separate chapter explains how mask is created.

Data Types which Nika can load

Igor native loaders

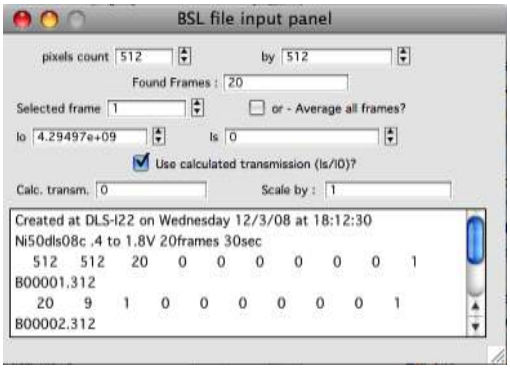
These are file types, which Nika can load natively. These can be loaded on any platforms:

tif	tif file
GeneralBinary	flexible binary data loader (see description below)
Pilatus	Should load NON COMPRESSED Pilatus data. Opens new screen with options. As of 1.66 can also handle compressed (BYTE_OFFSET) Cbf files.
Nexus	HDF based format used at APS, Diamond, … Used at 15ID SAXS. Contains a lot of extra information, which is stored in wave note for user use.
HDF5	under development at this time. Major headache as it is so flexible, that it is nearly useless. I have capabilities but no clear example of hdf 5 file being used in practice, so if you have such data, send me example so I can support it.
BrukerCCD	bruker SMART software for CCD
mpa	The software is MPA-NT (or just MPANT), version 1.48. It is from FAST ComTec, a German company that supplies multi-channel, multiparameter data collection and analysis tools. The hardware is the MPA-3 Dual-parameter multichannel analyzer (from FAST ComTec). That hardware provides the interface to multiwire 2D gas-filled X-ray detector from Molecular Metrology (recently purchased by Rigaku/Osmic).
DND/txt	This is loader specifically for data from DND CAT and APS. The data are tif files, but Nika will read text processing record, extract appropriate parameters and use those for reprocessing of the data. There is specific chapter in the manual on use of data from DND CAT.
mp/bin	mp binary format. for software above, Has header and binary data behind it.
mp/asc	mp format, ASCII column of data. Assumes squared shape (N x N pixels).
BSRC/Gold	BESSERC 1536x1536 Gold detector binary format. It has header and 16 bit binary data
DND/txt	Data from DND CAT (APS), located in original (beamline provided) folder structure. Find the text file containing reduced data, the code will open fully corrected tiff file from linked folder.
RIGK/Raxis	This is for Rigaku file “86” format. The code should be able to handle any size images, but tested was on 1k x 1k and 1.5k x 1.5 sizes. I tested the code against Fit2D and based the code on Rigaku C-code…. NOTE: known issue is with newer Raxis file format “100”. Need to rewrite this reader.
ADSC	binary file with header. Header contains “HEADER_BYTES” as first element.
WinView spe	Princeton WinView file format.
ASCII	ASCII matrix file format. If the file has extension “mtx” the code assumes there is accompanying file with extension “prm” which is read and parameters from that file are placed in the right Nika variables.
ASCII 512 x 512	one column ASCII data, present for 512 x 512 pixels.
Ibw	Igor binary wave format. This may be useful if the data are produced by Igor.
BSL/SAXS and BSL/WAXS	BSL/OTOKO file format. See http://srs.dl.ac.uk/ncd/computing/manual.bsl.html for description. Note: you need at least 3 files – file with name Xnn000.mdd which is header file, Xnn001.mdd contains one or more SAXS image4s with Xnn002.mdd calibration file, and/or Xnn003.mdd file with one

	or more WAXS images with Xnn004.mdd calibration file. The set is listed only one – with "proper" file name in the list of samples. Note, that without the inf file the loader fails and gives error. Read details below, please…
Fuji/imp	Fuji image plate reader (BAS2000 and BAS2500). Nightmare. Anyway, this image plate reader should be able to read 8 bit and 16 bit data from these two readers. There is a lot of variability of this code and I was unable to get test data for 16 bit images. Only 8 bits are tested to work at this time. If you have data from other settings, please, send them for testing to me.
ESRF/edf	ESRF ID2 "edf" file format. Should read also other edf formats but has not been tested. Reads ONLY files with ONE image per file – principally the format enables multiple frames (images) in one file, but this is really difficult to support. What to do with all those images???
FITS	This is one case of use of Flexible Image Transport System (FITS), R. J. Hanisch et al.: FITS standard, Astronomy & Astrophysics 376, 359–380 (2001)). It is likely going to fail on other files, but it is not clear how this is used in SAXS community at this time.
Mpa/UC	University of Cincinnati mpa file format.
SSRLmatSAXS	Format used at SSRL materials science SAXS beamline. Note, this instrument is supported under "Instrument support" where more helpful tool is.
TPA/XML	Tool used by Quokka, ANSTO, Australia SANS instrument. More support is in Instrument support.
GE binary	Used by GE area detectors.

BSL/SAXS and BSL/WAXS data format

Please note, that this "container" with up to 5 files has more information, which can be extracted. Therefore more controls are needed and separate panel opens up, see below:



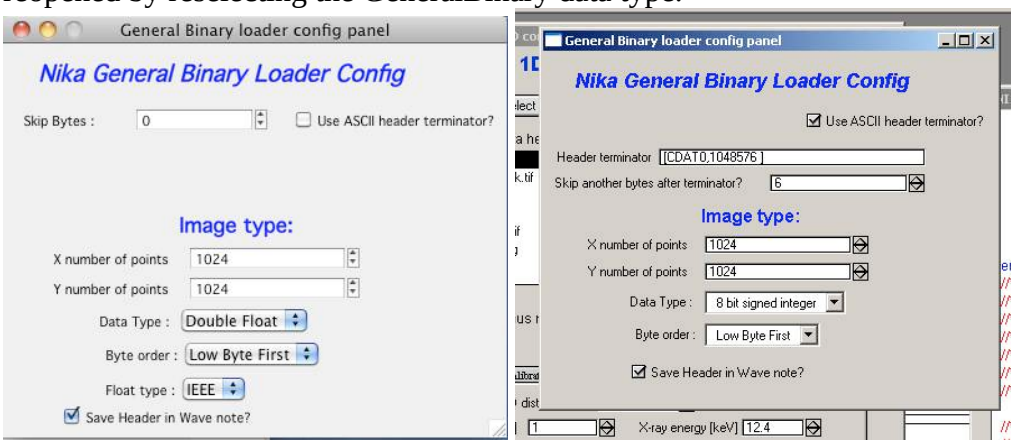
Following are loader details:
At the top are sixes of pixels for selected images, in this case the BSL/SAXS container contained images with 512 x 512 pixels. It contained 20 images, as indicated by "Found Frames" value. User can select to evaluate average of all 20 images (select checkbox "Average") or individual frames.
I0 and Is are extracted from the associated calibration file. In this case I0 has some value but there are no numbers for Is. I0 is ion chamber before the sample (incoming flux, monitor…) and Is are after the sample. Principally, when present, the ratio should be transmission, which is calculated for user and placed in "Calc. transm." value – note, this value is *always* Is/I0. Here it is 0, since Is is = 0.
In case the ion chambers had different sensitivity or Is is actually different detector type, user can also scale the Is/I0 by some ScalingFactor to correct for sensitivity.
Some of these values can be transferred to Nika variables:

Always the I0 is transferred to Nika calibration value for I0. User can use it by selecting the “Use Monitor?”.
On the BSL panel is checkbox “Use calculated transmission”? – when selected, value of (ScalingFactor*Is/Io) is transferred to Nika’s “Sample transmission” value for calibration purposes. It is used when checkbox “Use sample Transmission” is selected, see later Calibration discussion in this manual.

General Binary data loader

This is basically interface to Igor GBLoadWave function, customized for this application. Most parameters are copied from the GBLoadWave and so it is suggested user looks up this function in the manual for details.

Selecting GeneralBinary type on any window will pull up control screen. The configuration is same for all NIKA! The window can be safely dismissed (closed) and reopened by reselecting the GeneralBinary data type.



Top part:
Select number of bytes to skip – or check the “Use ASCII header terminator” if there is ASCII header with known ASCII separator before binary data. The known terminator (separator) in ASCII header- very useful, if the header bytes vary - input search term to look for. NOTE: Only first 40kbytes of data are searched, so the header cannot be longer than 40kbytes. For longer headers use the “Skip Bytes” option. Sometimes, you may need to skip few more bytes behind the terminator. Add those in the field below.

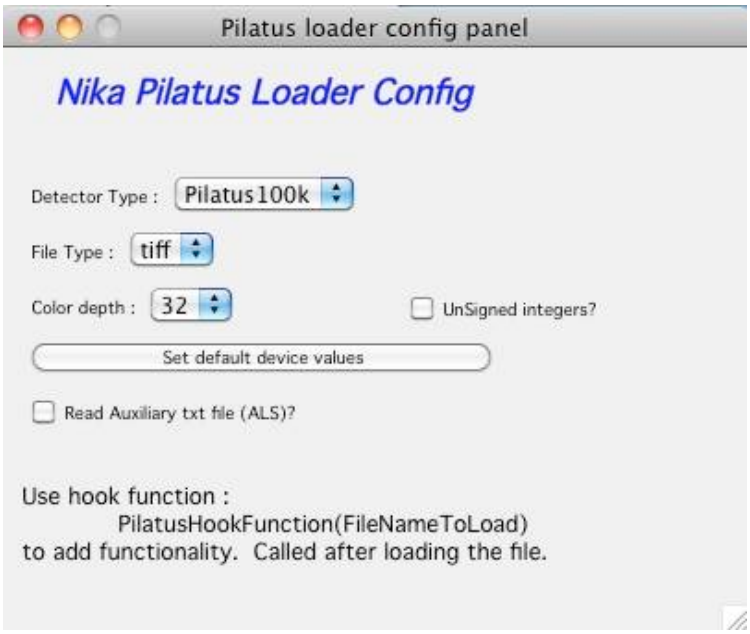
Image type part:
Size of image (rows x columns), type of data, for integer types byte order, and for floats type (IEEE or VAX), check the manual what this means… Save Header in Wave Note will add the skipped ASCII header to wave note, which gets propagated through the package into the final data… Very useful…

Other loaders with panels…

Some other loaders need user parameters to be included. There are two options, which are possible:

Panel. This is for example BSL/SAXS or BSL/WAXS format, where one file can contain multiple images. The panel enables selection of one or average of these images and provides some more information.
Function. Fuji image plate readers (BAS2000 or BAS2500) may be attached to hardware which saves data with varying “endiness”. A note is printed in history area which instructs user how to change endiness of the reader, if needed. This is set for the particular Igor experiment.

Pilatus



Can read: “tiff”, “edf”, “img”, “cbf”, and “tiff with floating numbers” (used for background subtracted images. Should manage 100k, 300k, 300k-w, 1M, 2M, and 6M files, but tested mostly 100k files (send files for testing, please). Can read Auxiliary txt files from ALS.
“*Set default device values”* sets pixel size to 0.172 mm which is at this time only pixel size available on this line of detectors.
NOTE: You can design function, called “PilatusHookFunction(“FileNameToLoad”)” which is run after each image is loaded and which can customize behavior of the reader. For example, the function can read the wave note - which contains the header from the Pilatus file – and extract usable information. Or it can read auxiliary text file and extract information there. It is called with the name of file currently loaded as string for parameter.

Using XOP to load

Only when MarIP (CCP4) xop is present:
MarIP/xop this will use xop to load MarIP data As noted above, this ccp4.xop can be downloaded from igorexchange.com (and compiled by user) or is available on request from me. Please, send me e-mail with platform you need xop for.

Fit2D data files

This loading has been deprecated and untested for years now. It is likely going to be removed in next Nika releases.
Fit2D can be used as "loader" for number of file types, which it can recognized. This can be done for now ***on Windows and Mac OSX version 10.4 and before****. Current Mac OSX version of Fit2D (14.101) cannot save usable tiff files on OSX 10.5 and unless I make changes in my code, Fit2D cannot be used as loader.*
Fit2D method is somehow cumbersome, basically I create script file, which starts Fit2D, loads file in Fit2D and saves it immediately as tiff file. This tiff file is then read by Igor and immediately erased.
Note, that since I do not have these files to test, I cannot really verify function of all of these files… I have tested Mar image plate for now and it works… If you find one that does not work, please send me test file and I will fix the bug.

ADSC	*ADSC Detector Format : Keyword-value header and binary data*
Bruker	*Bruker format : Bruker area detector frame data format*
BSL	*BSL format : Daresbury SAXS format, based on Hamburg format*
Diffract	*Compressed diffraction data : Compressed diffraction data*
D*IP2000*	*DIP-2000 (Mac science) : 2500*2500 Integer*2 special format*
ESRF	*ESRF Data format : ESRF binary, self describing format*
Fit2D	*Fit2D standard format: Self describing readable binary*
BAS	*FUJI BAS-2000 : Fuji image plate scanners (aslo BAS-1500)*
GAS	*GAS 2-D Detector (ESRF) : Raw format used on the beam-lines*
HAMA	*HAMAMATSU PHOTONICS : C4880 CCD detector format*
IMGQ	*IMAGEQUANT : Imagequant TIFF based format (molecular dynamics)*
KLORA	*KLORA : Simplified sub-set of "EDF" written by Jorg Klora*
MarIP	*MAR RESEARCH FORMAT : "image" format for on-line IP systems*
MarPck	*MAR-PCK FORMAT : Compressed old Mar format*
MarIP	*NEW MAR CODE : Same as MAR RESEARCH FORMAT*
PDS	*PDS FORMAT : Powder diffraction standard format file*
PHOTOM	*PHOTOMETRICS CCD FORMAT : X-ray image intensifier system*
PMC	*PMC Format : Photometrics Compressed XRII/CCD data*
PRINC	*PRINCETON CCD FORMAT :X-ray image intensifier system*
RIGK	*RIGAKU R-AXIS : Riguka image plate scanner format*

Calibrated 2D data files

EQSANS (ORNL) – text file with four columns – Qx, Qy, Intensity, Uncertainty, map of 400x400 points. Generated by EQ SANS instrument at ORNL (provided ~ 5/2014).
canSAS/Nexus – as of 6/2014 this standard is under development. My best guess what the format will be. Hdf5 file. Contents structure varies. For details and examples see http://www.cansas.org/formats/canSAS2012/1.0/doc/

Invert 0,0 corner

As default Igor displays 0,0 of the image in the top left corner. This seems to be distressing for some users, so if checked, images will have 0,0 in the left bottom corner. Nothing else is changed, so the orientation of sectors WRT original image is preserved and reduced data are the same as without this checkbox checked. Simply, the processing of Nika package is independent of this checkbox, it is ONLY cosmetic…

Sort order

Decides how the data are listed in the listbox.
Four options:
None – list as provided by OS.
Sort – the old method. Alphabetical (but numerical order may get wrong)
Sort2 – alphabetical, but taking care of sorting out smaller number before larger ones.
001. – this one assumes, that end of file name, before extension, is number. Before number you need to have "" and after number must be "." Followed by extension.
Invert _001
Invert Sort
Invert Sort2
All inverted sorting simply reverses the sorting logic.

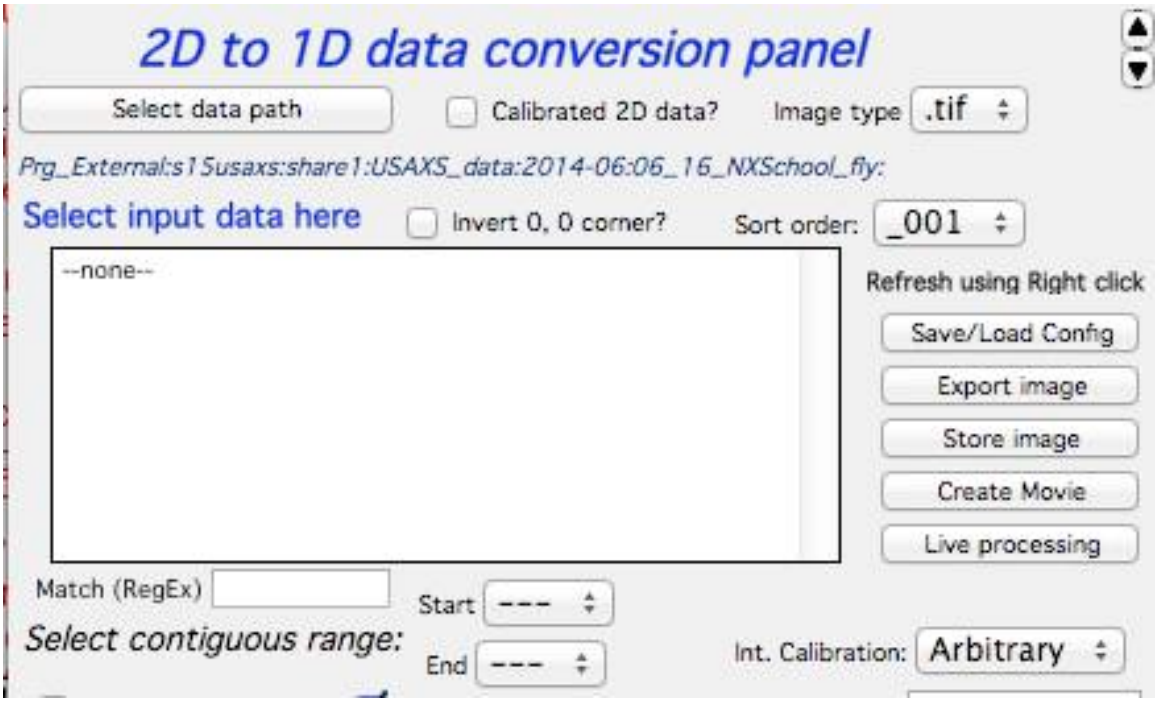
Try them and see, which works best for you.

Match

Using RegEx now. This is Grep language using regular expressions, very powerful. For simplicity: match names containing (anywhere) test, just type in this field test. To match names starting with test type in ^test. Names ending with tif can be matched by tif$ and so on. Note that to match any single character you need to use .

Side controls

There are few controls next to the Listbox where user can select the data:

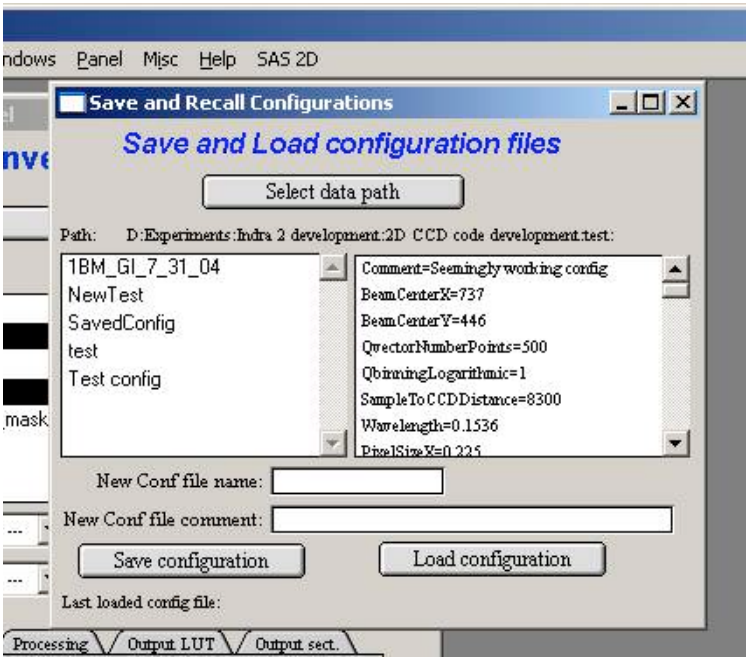


Refresh:

This button was removed in 1.66. The refresh and some other functionality was added to right click for most Listoboxes in Nika.

Save/Load Config

"**Save & recall config**" will allow user to save current settings – or load saved settings- in the tabbed area as "configuration". User can save the configuration file in any place on the hard drive he/she wishes – ideally with the data!
Following dialog is presented:



Explanation of controls:
"Select data path" – select path to folder with the configuration files
Path is displayed below
Left window – shows the names of configuration files found in this location
Right window – shows content (note, first line is user comment) in the selected configuration file. If new configuration file is selected, content of the first one is shown.
New Conf file name – User input for new configuration file to be created
New Conf file comment – place to store info about what this conf file contains!
Save configuration button – save current setting of the tabbed area
Load configuration button – load stuff from config file into the program. Note, your current configuration will be overwritten and there is no way back, if you did not save your configuration…
Note: names of dark field, empty beam, mask, and pix2D sensitivity are not saved and are not reloaded, when configuration is reloaded. This would really be very complicated…

Note, that the config file has name and Comment string. If you get lost altogether, you can also see on the right hand side what values are stored in that configuration file. It is really good idea to use meaningful names and comments – especially if you have a lot of configuration files ☺

Export image

Enables user to export the main 2D graph as tiff image from Igor.

Store Image

Enables user to store the current main 2D image in Igor Experiment for reference… Remember, they can be large and so do not store too many or the Igor experiment may

become unmanageably large. Also, there is not much support for dealing with these images (it is not really clear what user would want to do with them to me), so you are on your own and use Igor tools to handle these images…

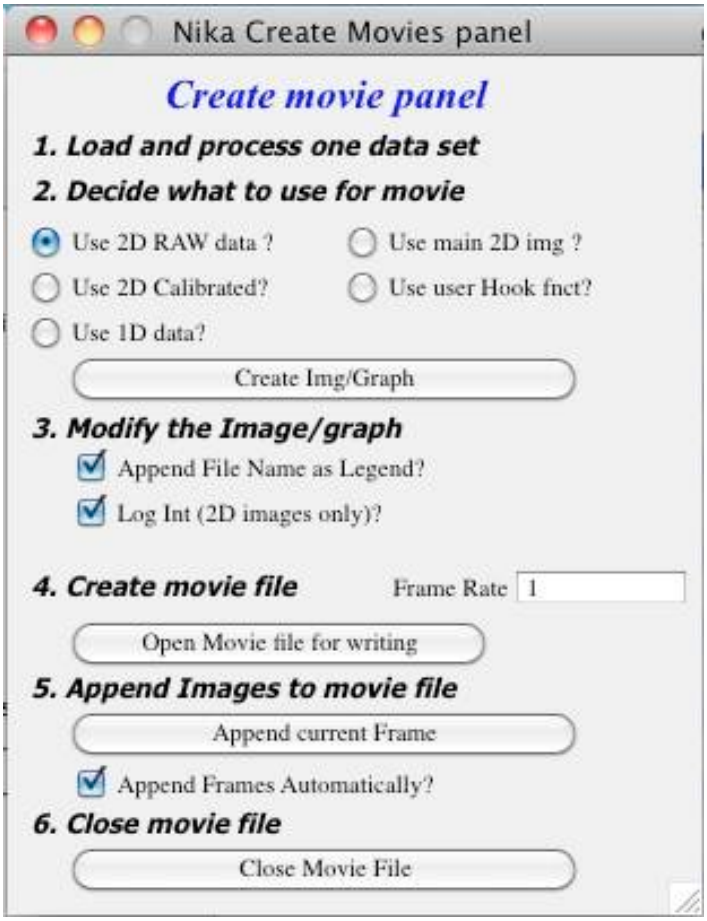
There is NO WAY to load these images back into Nika at this time. It can be done manually, but not through Nika menus & functions.

Create Movie

This opens panel, which is interface for ability to create movie from either 2D images or 1D lineouts…

Note: The way this tool is designed, image is added every time any of the Convert butons is called. It is possible to use this tool to create movie from RAW data only manually (by manually adding each frame) when user wants to load the data inusing the "Ave & Display sel. files" button. Using this button you cannot use Automatic add function.

But the same result you can achieve by using convert buttons and simply not converting the data any way (not checking anything on "Main" tab and any of the "Sectors", "Prev" and "LineProf" tabs. In this case you can add frames automatically, which is probably more useful…



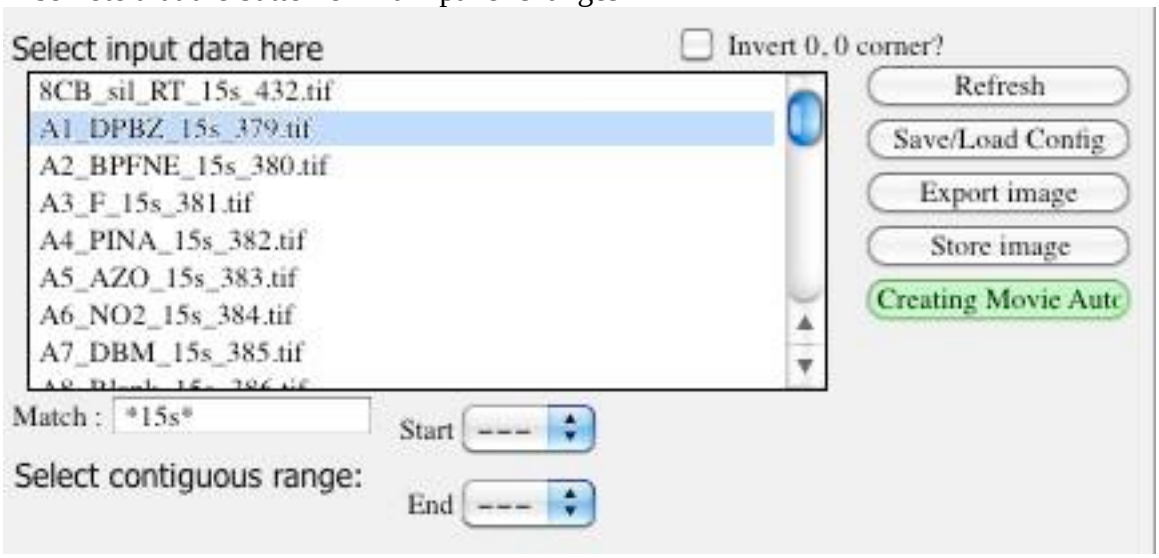
The panel walks user through the steps necessary…

1. **Load and process one data set.** In order to use this feature, user needs to first load some test data set (image) to have test case to try the display options. This test case needs to be processed all the way needed…
2. **User needs to decide what to actually add into the movie**… Selecting the checkboxes creates the image, if you need to you can recreate (or pull up if it is hidden) the image by the button "Create Img/Graph". There are various options:
    a. 2D RAW data image. This image is using separate image, copy of the RAW data. The graph can be customized by user (zoom, range scaling,…). Since the code for subsequent images replaces this separate copy of the image with newly loaded the wave, this does not modify the image itself. Therefore the display should be relatively stable and under user control – it should stay as user zoomed/set color range/etc...
    b. 2D Corrected data image. Same as above, but the image used is a copy of the fully corrected 2D images (empty/background subtracted, calibrated…). Again, the controls are left to user present ones since the code overwrites the separate copy of the data and therefore swaps the new data into the image without major recreation. Should be relatively stable without major changes to the way the data are displayed.
    c. 1D data. This is graph of the lineouts created by the code. NOTE: if you are creating more than one lineout from each image (like when using multiple sectors), all of these may be subsequently used! This may be good (movie of sequence of sectors on one image) or bad (for movies from many images). There is no way of skipping and using only specific sectors. Use Hook function to create that…
    You have relatively lot of controls of the graph, same as in the above options 1 and 2, as the data for this graph are a separate copy of your last data. When the old ones are overwritten, the new ones are "swapped" into the graph and replaced without modifications to the graph. So the graph should stay without major changes, unless set that way. For example, if Axis are set to auto scale, they may change. But if they are set to fixed start/end, they will stay fixed. At least I hope ☺.
    d. Use main 2D image. This one simply uses the Main 2D image. Seems very good choice - BUT: that image is recreated every time from scratch so there are very few controls available to user – you either like it and then use it, or you cannot use this method. You should, of course, use the controls on main panel to modify the image – like use RAwor Processed data, display sectors, beam center, colors, or Image with Q axes… That works, but you cannot control other things, such as zoom range etc.
    e. Use user Hook function… This is advanced method. Here you can do whatever you want to create the image you want to append, just call the function: Movie_UserHookFunction and if it exists, it will be called. This function MUST generate graph/image and leave it as the top image. This top image is added to the movie when called… Note: while this is advanced programming, this is way to get really what you want into the movie… Below is commented out example (present in the code also)

which pulls up the main 2 D image and prints a note. You can add here any other formatting which you want to do or use…

```
//Function Movie_UserHookFunction()
//		DoWindow CCDImageToConvertFig
//		if(V_Flag)
//			DoWIndow/F CCDImageToConvertFig
//			AutoPositionWindow  /M=1 /R=NI1A_CreateMoviesPanel CCDImageToConvertFig
//		else
//			Abort "Main 2D windows does not exist"
//		endif
//		print "called Movie_UserHookFunction function"
//end
```

3. **Modify the Image/graph**. Here you can modify some of the appearance of the image/graph. If you want to display log of intensity in the images, here is your only chance (for first two options). You can append also file name – and edit the appearance of the legend manually – as long as you do not change the reference to global string, which contains this name, you can change font, size, location…
4. "***Open movie file"*** button. You here create movie file and open it for writing – external file for Igor experiment. Remember to set proper frame rate. Frame rate of 1 is 1frame/second, 10 is 10 frames/second. So if you have 100 images to add, at 10frames/second the whole movie will play for 10 seconds.
   You can have ONLY one movie file opened at one time (Igor limitation). The button greys out when movie file is opened.
   Also note that the button on main panel changes

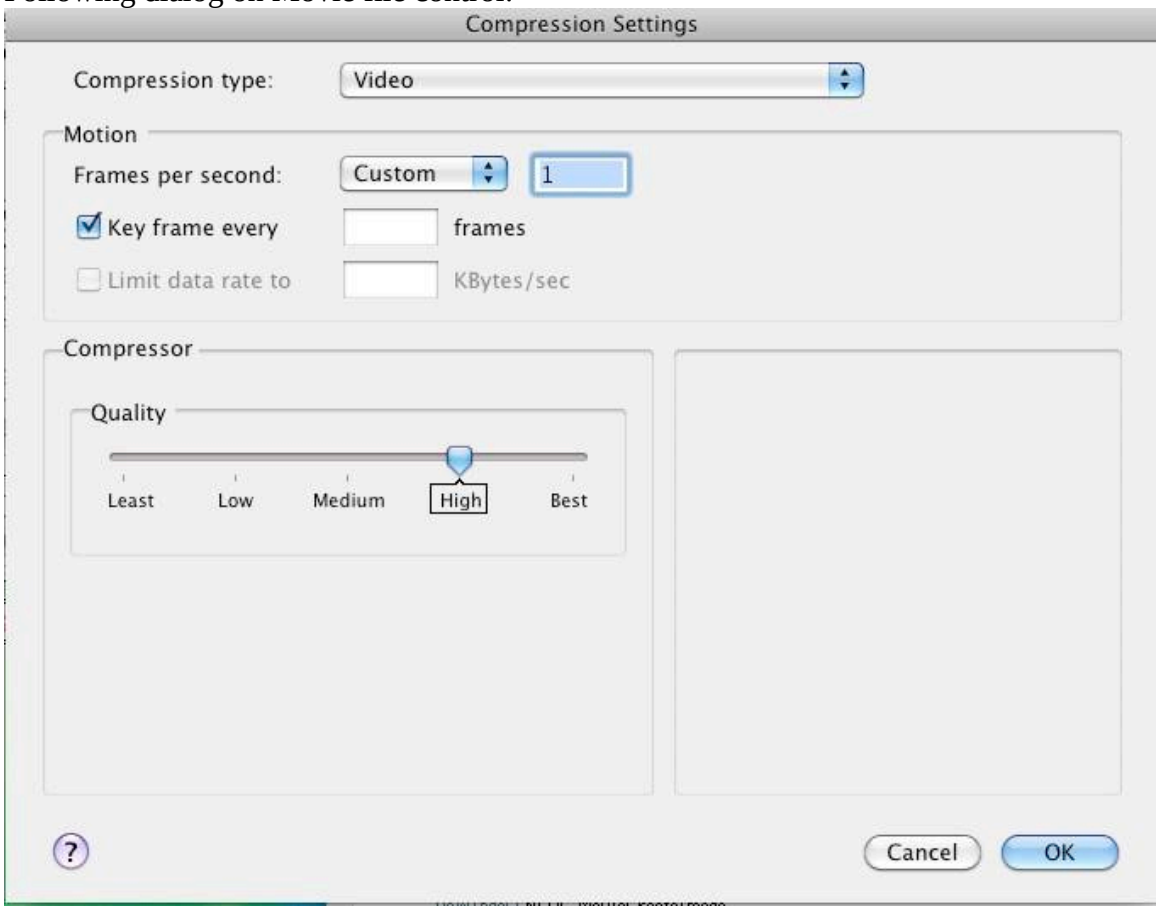


5. **Append Images to movie file**:
   You have two options:
   a. "*Append current Frame*" button. – Works always, appends current image/graph per selection (see above item 2) manually to the movie. Use when you want to control the appending of the frames really well.
   b. Checkbox "*Append Frames Automatically*" – if set, after loading & processing every image a frame is appended automatically.

6. *"**Close Movie file"*** button. Well, before you can play it, you need to close it…

Warnings: It is very likely all hell breaks loose if you close Igor experiment and reopen it later with Movie file opened for writing. I suspect Igor will close the movie file on file close, but Nika will NOT know about it. While it is principally possible to fix this in the code, there are good reasons why not to do it. So keep this in mind and do not leave the Movie files opened when closing the Igor experiment. At least close the Movie file before you try to add any frames to it.

Following dialog on Movie file control:



Is Igor panel and here are your last chances to control what and how it gets created… I have limited information on what works best, so try this your self… Keep in mind, that while on PC you can create either mov file (Quicktime) or AVI file, it may be challenge to get avi files play on Mac. I suspect that considering the avi mess in video formats, you may have much better chance to play QUicktime movies (mov)… But there is no guarantee on unknown machines, that they will have Apple quicktime.

Note, that every time Nika adds frame to the movie, it prints in the history area: "Added frame with data : xxxxxxxxxxxx.tif to movie". This tells you what you added…

Live processing

Live processing is attempt to make automatics display or processing data for instruments at synchrotrons or neutron sources. When pushed, it opens new panel:

The description is hopefully clear. You can start background process, which is sleeping for the "Update time". If Igor Pro is not busy at the time when woken up, the background process will basically run "refresh" command and if new file is found (after applying all Match RegEx and Data type matching, this new image is automatically processed using the settings in Nika.

Note, that user interactions may delay this processing, so if user is using Igor, this may not happen. However, if user is using sporadically this update may happen at inconvenient time, so make sure if you want to "Play" with the file you stop this background process.

Note checkboxes: "Display new image" or "Convert new images", which control, which button is pushed by this tool when new image is found. The first pushes "Ave & Display sel. file(s)" while later pushes "Convert sel. files 1 at time".

Intensity calibration

Most of the time the data in Small-angle scattering are normalized and not calibrated. This prevents users from obtaining quantitative information about volumes of scatterers and specific surface areas (etc…) using data analysis packages (such as *Irena*). If users collect standard sample (e.g., Glassy carbon: ***Zhang, F., et al., Glassy Carbon as an Absolute Intensity Calibration Standard for Small-Angle Scattering. Metallurgical and Materials Transactions A, 2010. 41(5): p. 1151-1158.***) the data can be put on absolute scale – either $cm^{-1}sr^{-1}$ (volumetric calibration, also $cm^2/cm^3/sr$ – typically shortened as cm^2/cm^3) or cm^2/g for weight calibration. The popup :

enables users to select which units of absolute intensity calibration they want to write in wave note of the data. Other packages (Irena) may use this information and then it may be critical to have the right one in there.

Name trimming

Igor Pro has 32 character limit for names but many operating systems allow much longer names. Also, users are notorious for using file names as abstract.
If Nika is suppose to save the data in Igor experiment, it needs to cut the name down to smaller size – and since it is using part of the name to describe how the data were reduced, it limits user useable length of the string to 20 characters…

In these controls user can select how to handle too long file names – remove part of the name (string) – and if still too long, trim start or end of the remaining string…
Here is example:
Name My_Name_is_SIMPLYTOO_long_for_comfort_even_with_removal.tif
55 characters. Perfect.
Trim end would result in name: My_Name_is_SIMPLYTOO
Trim start: comfort_even_with_removal
And remove “SIMPLYTOO_long_for” and trim end : My_Name_is__long_for

Etc…

Bottom controls

These controls have following functions:
"**Ave & Display selected file**" will average all selected files, which are selected in the list box, and display them as one figure. The program will just load and display the CCD images, including some processing (dezinging), if selected.

Note, if more than 1 image is selected, the images are first AVERAGED – that is intensities for each pixel as summed together and then divided by number of images.

"**Convert selected files 1 at time**" will load one after another the files selected in the list box and process them according to selection in the tabbed area.

"**Ave & Convert selected files**" will average all selected files in the list box and process them according to selection in the tabbed area.

Note, if more than 1 image is selected, the images are first AVERAGED – that is intensities for each pixel as summed together and then divided by number of images.

"**Save displayed image**" will save displayed image into tiff file for future use. This is method, how to for example average number of images and save them for single empty or blank image.

"**Skip Bad files**" Enables to skip automatically processing of files, which have too low intensity (SetVariable control with limiting value appears when selected). Used to skip files which were accidentally NOT exposed in case of failing shutters or other issues.
"**Display RAW data**" will display in the image right of the panel the UNCORRECTED data file as loaded in. Values for the pixles are raw counts from the detector.
"**Display Processed**" will display in the image right of the panel the fully CORRECTED and CALIBRATED data. The values for the pixles should be directly absolute intensity in this case. This choice is not available, if image was loaded through using "**Ave & Display sel. Files(s)**". In this case no processing of the image was done. Use button "**Convert sel. Files 1 at time**" or the other buttons…. Just remember, that only the last image is available for display.

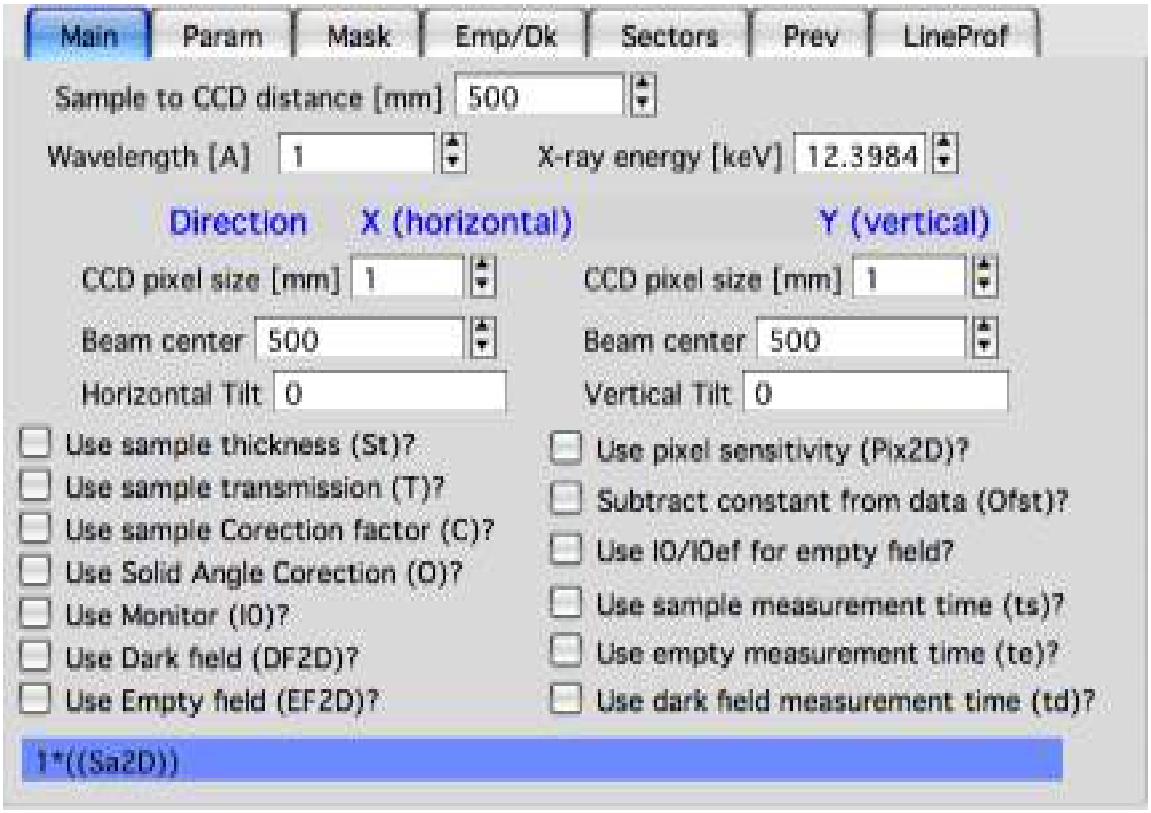
"**Display beam center**" will add circles in the image showing where beam center is set
"**Display sectors/Lines**" will add lines showing sectors or lines, which are selected for data analysis (if any)
"**Log Int display**" will switch displayed image into log (intensity) or linear (Intensity).

"**Image with Q axes**" Appends Qx/Qy (or Qz/Qy) axes to displayed image. Note, when unchecked, it has to recreate the image, since these Q axes cannot be removed any other way.
"**Image w/ Q axes with grid**" Appends Qx/Qy (or Qz/Qy) axes to displayed image – with grid lines. Note, when unchecked, it has to recreate the image, since these Q axes cannot be removed any other way.

Geometry/processing controls

Note, that if images are averaged, they are first averaged during loading, and then – during processing to create lineouts / square matrix are corrected as described below. Therefore all parameters here related to single (if possibly averaged) image!

These are controls in the tabbed area.

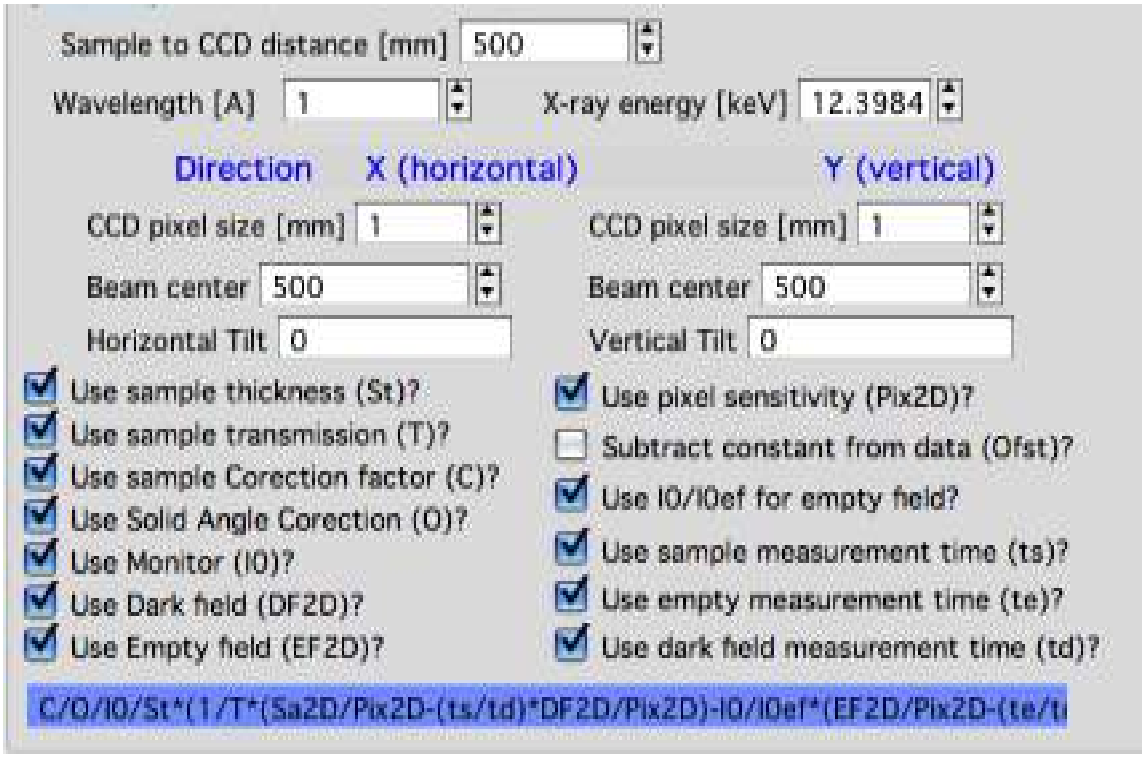


We will now go through each tab separately

Main

Here are some very clear parameters, related to SAXS camera geometry:
Sample to CCD distance in millimeters, Wavelength/Xray energy (these windows are linked), CCD image pixel size in mm (in X and Y directions). Note, X direction is horizontally, Y direction vertically. And Beam center position. Note, one can display beam center (to check it) in the graph by checkbox below the tab area.

And further there is pile of checkboxes, which describe method how to calibrate the data. Note, that formula used for calibration appears below to avoid any misunderstanding of the method. Select method needed for processing – and following tabs will have the appropriate controls available.

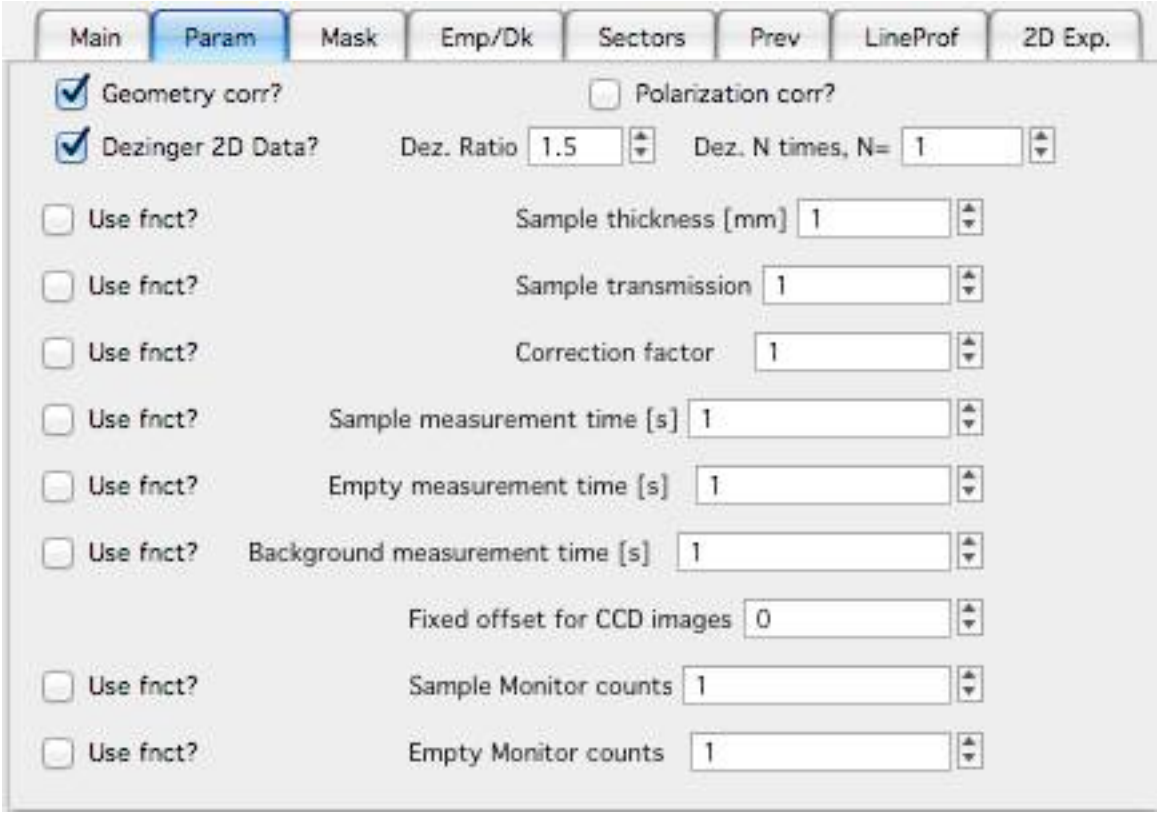


Note, that "**Use of Dark field**" and "**Subtract constant from Data**" cannot be used at the same time (they are effectively the same type correction)…

Note, only the appropriate controls will appear, so seeing all of these at the same time should be VERY unusual…

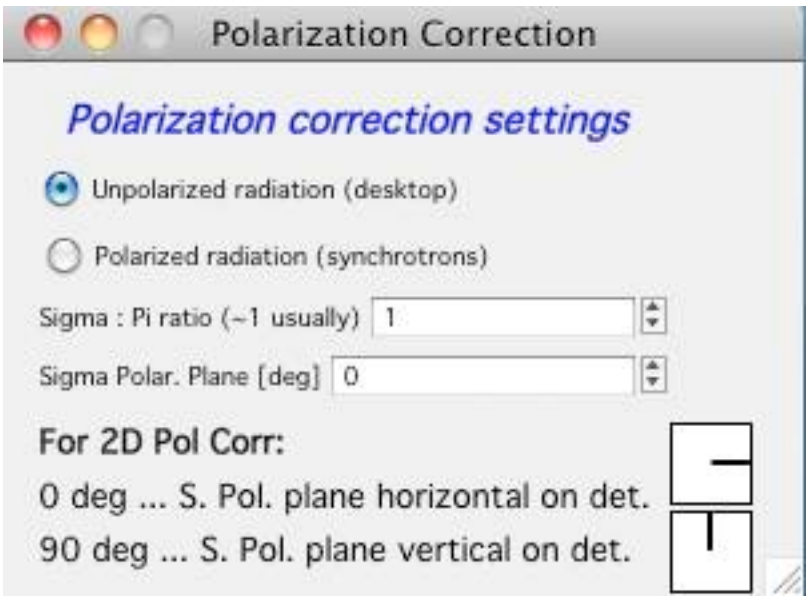
Comment for Use of **Solid Angle Correction**: When selected, the data are divided by solid angle of the central pixel (same value for all pixels). To correct for change in pixel solid angle as function of scattering angle, use Geometrical correction. Most of the time we do not bother with this option – if you use secondary calibration standard (like Glassy carbon or water) solid angle correction is included in the Calibration constant. If you do not use calibration and have relative data, you do not care also. The real need for this option is when you use data obtained in different sample to detector distances and want to combine the data together. Then this is necessary option.
Just remember, if you have obtained calibration constant, it is linked with the choice of the Solid angle correction.

Param



Here are standard controls (self explaining I hope):
"**Geometry correction**" – fixes the **VARIATION** of solid angle projection of the pixels on planar CCD detector. Mostly negligible for SAXS data… Just for completes, this divides the intensity at each pixel by (cos(2Theta))^3. And for those, who do not understand this formula, it took me may be 3 weeks to check it (I stole it from NIST data reduction). Very simplified, one cos(2theta) corrects for change of pixel radial direction as function of scattering angle, second cos(2theta) comes from change in distance between sample and detector as function of scattering angle in radial direction, third cos(2theta) comes from the same correction for tangential direction. Tangential size of pixel does not change as function of scattering angle.

"**Polarization Correction"** – Correction for either unpolarized radiation (desktop instruments with tube sources for example) or for Linearly polarized X-ray sources (synchrotrons). Opens up a new panel.



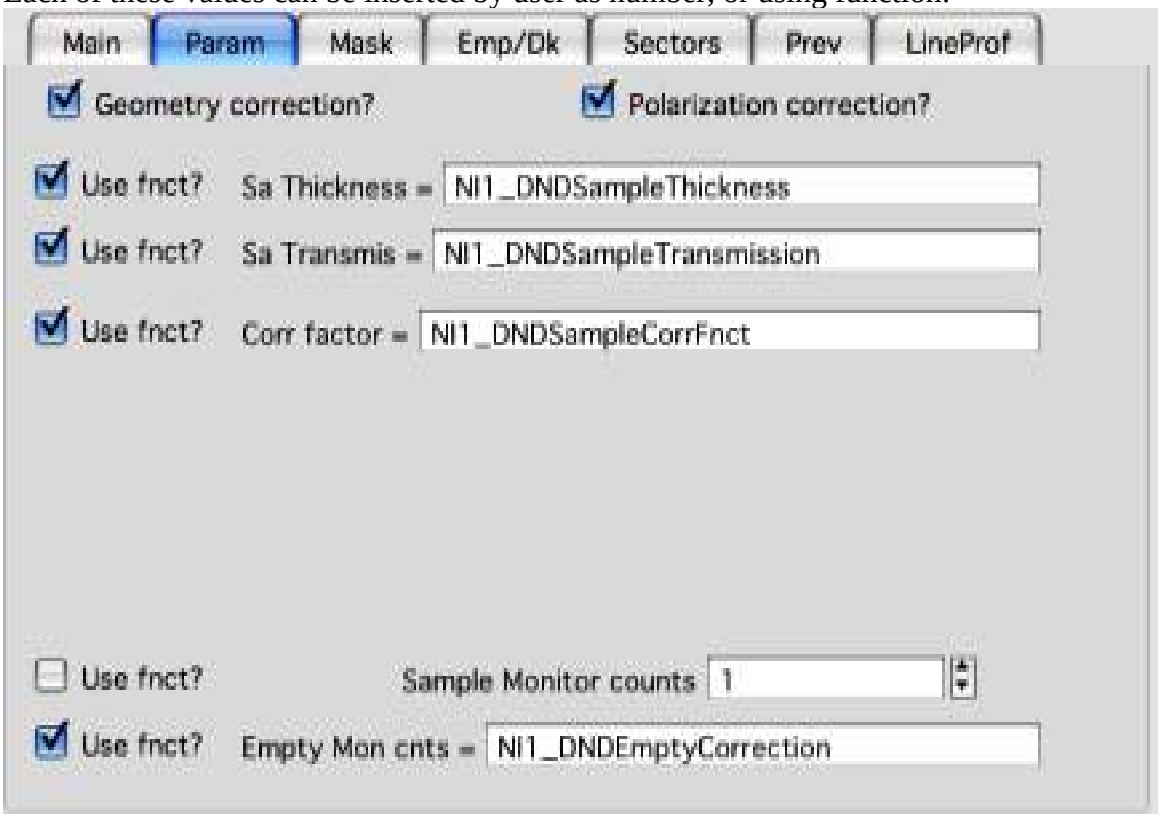
For unpolarized radiation use “Unpolarized radiation”. This is applicable ONLY to unpolarized radiation, the intensity data are corrected by formula:
Intensity_corrected = Intensity_measured / (0.5*(1+cos((2theta))^2))
For linearly polarized radiation use “Polarization radiation”, see separate chapter on Polarization correction little bit further in this manual.
By the way, for small-angle scattering each of these corrections is negligible.

“Dezinering” - Data, Empty, and Dark field images can be “dezingered” during loading. In this procedure each point is compared to surrounding pixels and if it is significantly larger (that is the dezinger ratio, if 2 then if the pixel is 2x larger than average of surrounding pixels) it is replaced with the average of the surrounding pixels. This is to remove spurious very high intensity points, which occur on some instruments.

It is possible to dezinger each image multiple times, in case the “zingers” are larger than single pixel.

Calibration/processing parameters:
Sample thickness in millimeters, **transmission** as fraction.
Correction factor is for secondary calibration factor.
Measurements times in seconds, for each image.
Sometime one wants to use measurement time to correct images collected at different time exposures. While not suggested, it is possible to do here. I strongly discourage this.
Monitor counts allow scaling data by using monitor on incoming intensity.
“Fixed offset for CCD images” this is single value to be **subtracted** from each pixel of image to be processed.
“Monitor counts” use monitor counts to scale images (Sample/Empty)… This makes no sense for dark field…

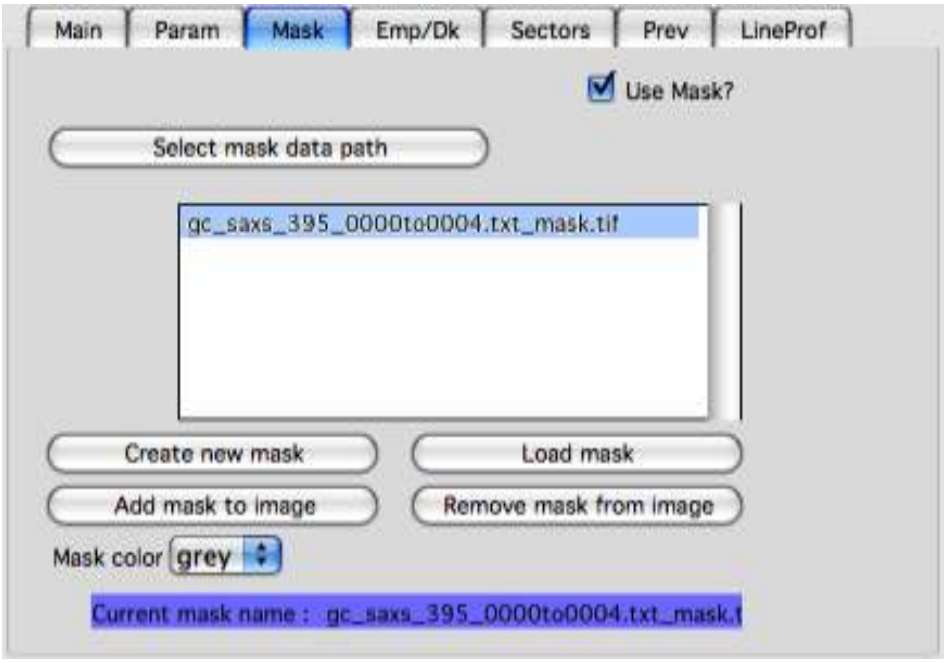
Each of these values can be inserted by user as number, or using function:



These function need to be “look up” functions, which are called with image name as parameter (FunctionName(“ImageName”)) and must return single real number. The real use is to provide automatic look up of parameters from some records written by instrument. Above example is from included special support for DND CAT instrument.

Let me point out once more here, that using some of these corrections together makes no sense… Choose wisely.

Mask



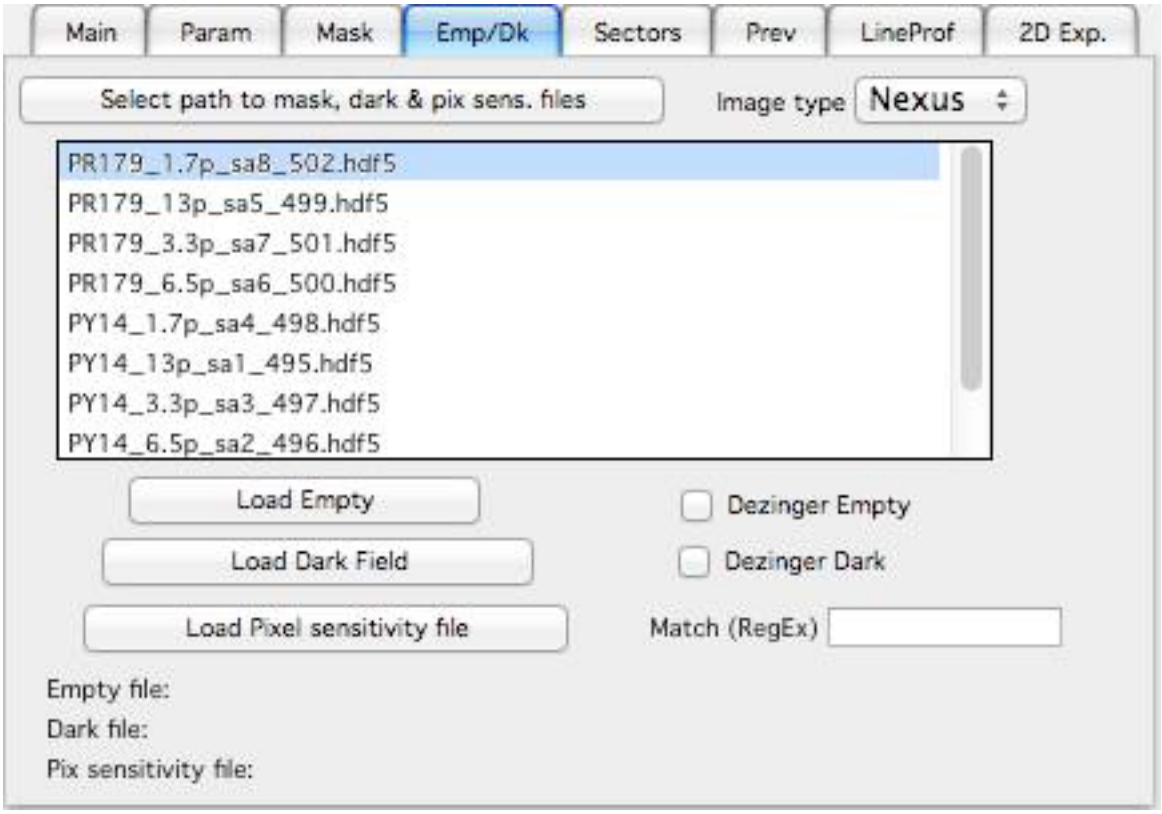
First checkbox, if Mask should be used (did not fit on the front tab…), button to select path to files with masks. Note, mask files created by Nika used to be always tiff files, with name in following manner: UserName_mask.tif Starting with version 1.49 they are now hdf5 files. These can be loaded in same as tiff files, but have anb advantage that these can be later modified in the mask tool…

Following are function of the buttons:

1. Create New mask – calls tool to create mask (see later in the manual)
2. Load mask – load file selected above in the list box as mask
3. Add mask to image – adds mask into the 2D image from the image
4. Remove mask from image – removes the mask from the image

Mask color – allows to change color (red, green, blue, black) of the displayed mask…
Current mask name – shows name of last loaded mask file

Emp/Dark

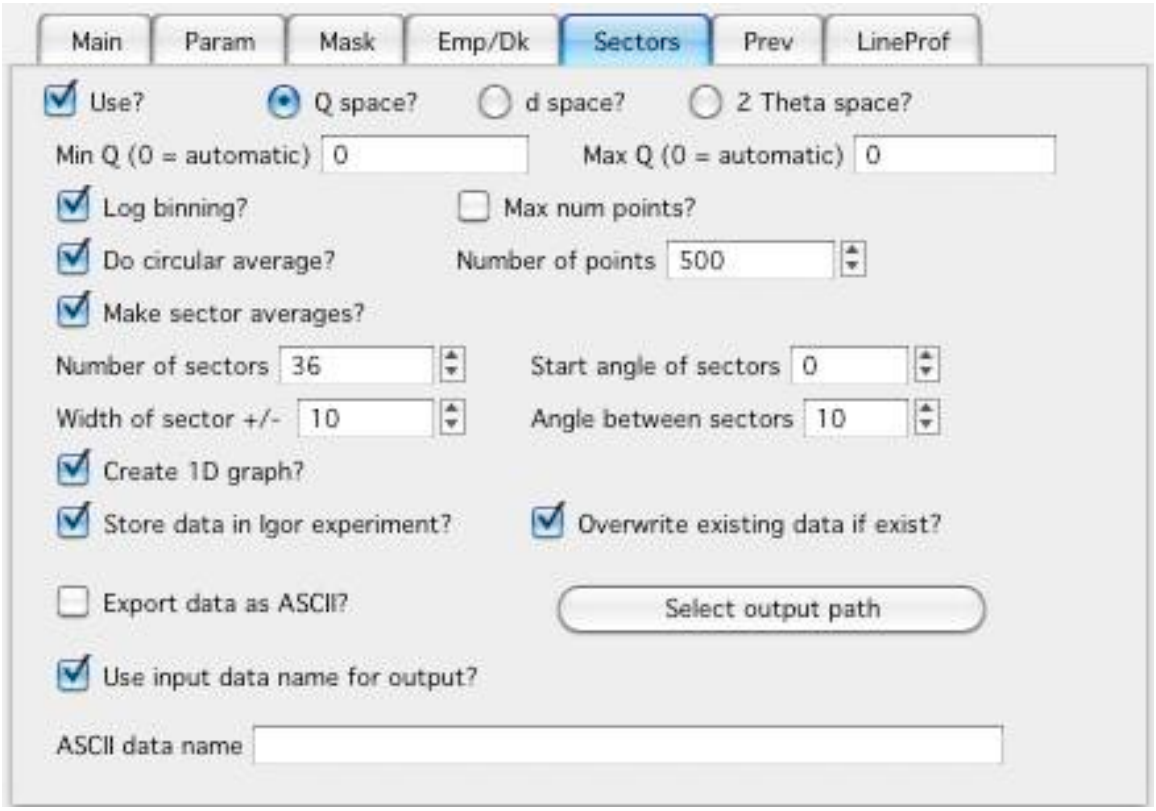


Here are controls for Empty/Dark field/pixel sensitivity (aka flood) images.

Button “Select path to mask, dark & pix sens, files” Selects path to data with the Empty, Dark field etc. I believe the files need to be the same type as data file (I need to check this).

Further buttons load the Empty/Dark/Pixel sensitivity, allow Dezingering of these (same method as the sample dezingering as selected above). And at the bottom are listed the file names of the files loaded…

Sectors



This tab controls how data are processed when method using " reverse Lookup tables" is used. This is the more suggested method for regular data processing. In this method Nika creates first lookup table for each sector defined and then can process much faster subsequent data files with the same geometry…

Controls:
Q space/d space/ 2 theta space – Output as function of Q, d, or 2 theta…
Min/Max (Q, d, 1 theta) range of evaluated Q, d, 2 theta. Set to 0 for automatic – automatic means, that the min/max is set for first q/d/2 theta which has non zero intensity
"**Log binning**" – check yes if Q/d/2 theta binning should be in logarithmic.
"**Number of points**" – number of points in Q/d/2 theta which should be created.

Do circular average – self-explanatory.
Make sector averages – do sector averages. Controls below control orientation and sizes of sectors. To see how the sectors are places, check the checkbox at the bottom of the control panel.

Create 1D graph – if checked, 1d graph with output is created (if necessary) and data added. Note, the graph may be crowded very fast, since data are added, and added…

Store data in Igor experiment – keep data (as qrs triplets) in current Igor experiment. Overwrite existing data if exist – if data with the same name exist, overwrite without asking. Otherwise, you will be asked.

Export data – export ASCII data
Select output path – select where data are to be placed.

Use input data name for output – automatically name 1D data (with sector information added as DataName_Angle_width) by input data name.

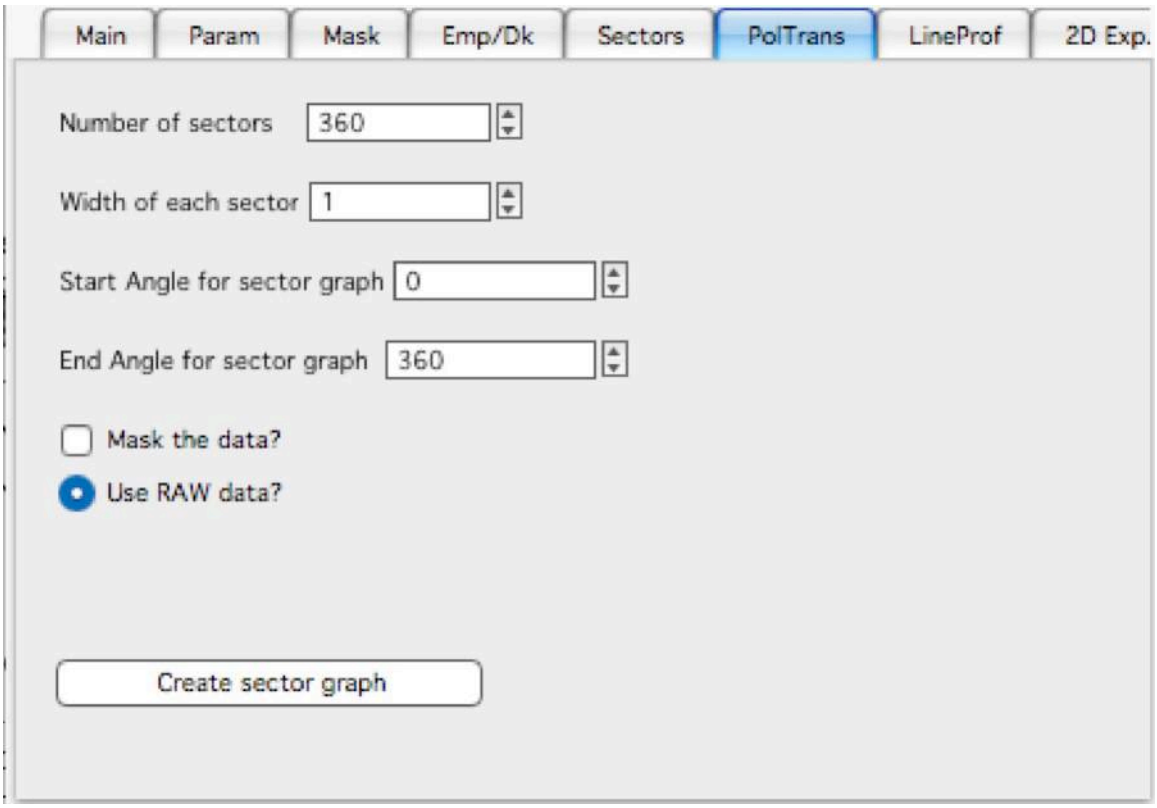
ASCII data name – if the above is not selected, this is place to place name for output file. Note, if there is nothing available for the code as sample name, it will ask for some…

PolTrans

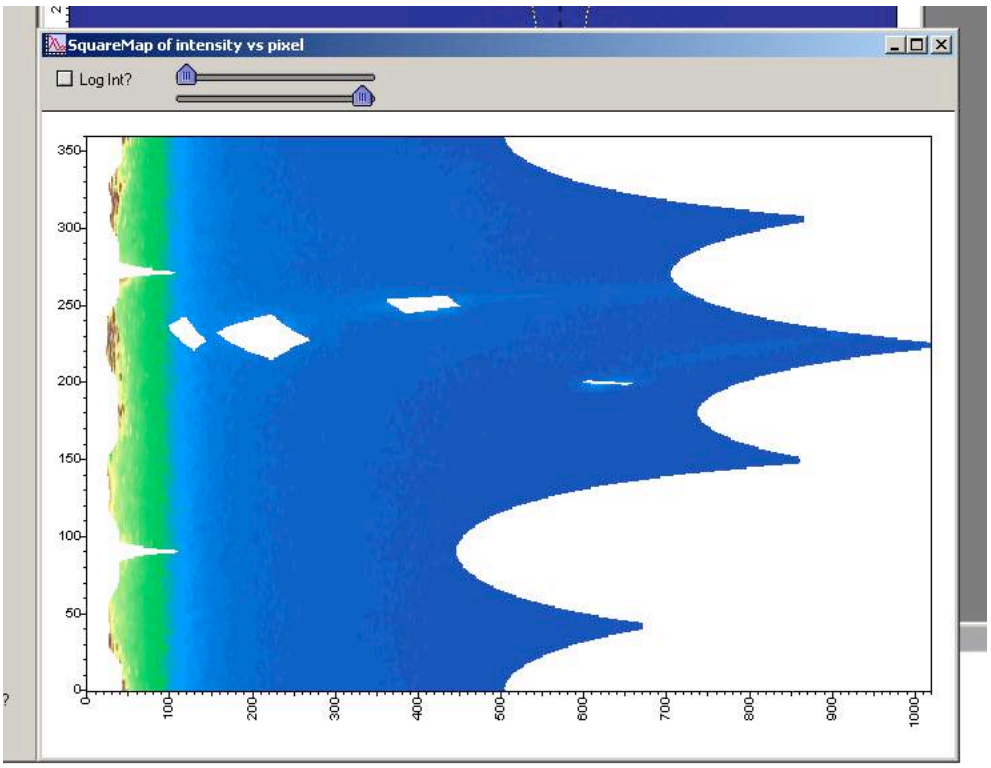
This means: ”Polar transformation” – prior (pre 1.68) name was “Preview” which is the intended use of this tool…

First:

This tool can use the calibrated data set (as well as RAW data set, depending on checkbox setting) so same calibration procedure is used as for the other processing. This tool is, however, less precise and does NOT produce useable errors. Be warned, this tool is meant as quick look on the data in different directions and not for final data processing…

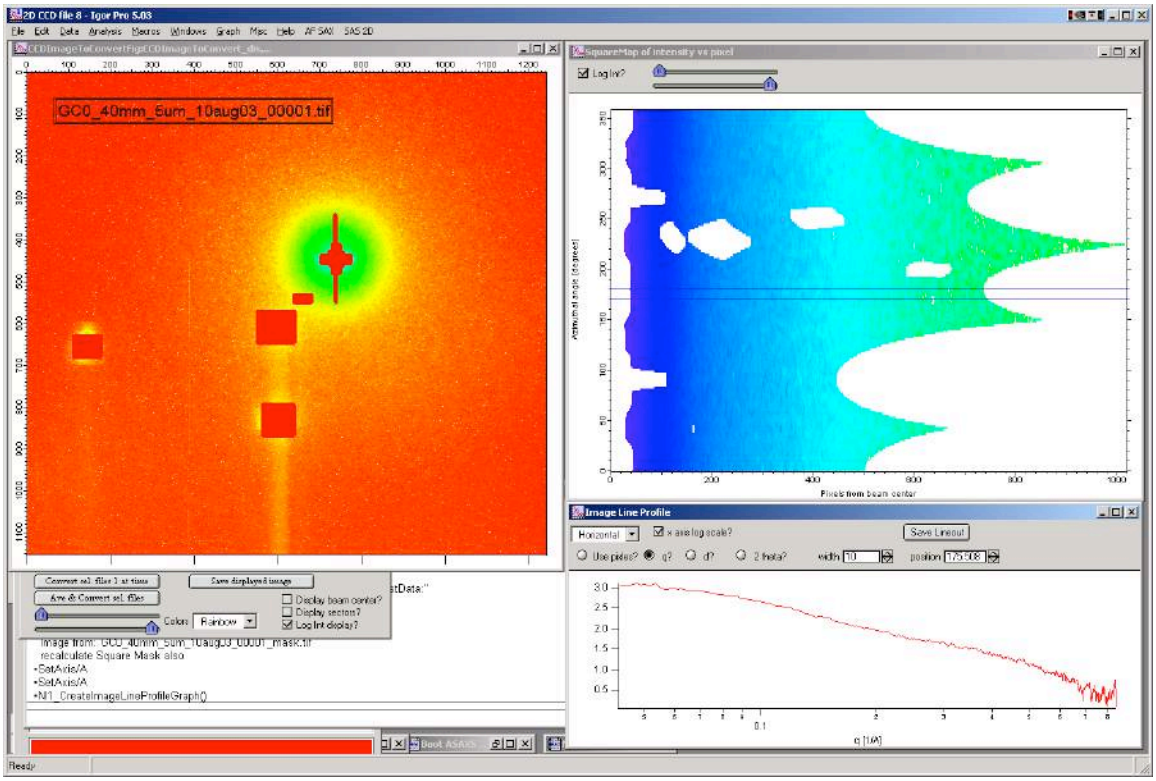


This method is used to convert Intensity vs azimuthal angle from “polar coordinates” around beam center to plot where azimuthal angle is on vertical axis, pixel coordinate is on horizontal axis and intensity is expressed as color map. In here, one can produce rectangular graph:



On vertical axis is angle from 0 degrees axis (horizontally right from the beam center) and on horizontal axis is pixels distance from beam center. This is effectively set of lineouts in all azimuthal angles. It should be noted, that the code works very well for relatively small widths – may be up to 5 degrees, then the code becomes less precise, so keep angles small. Suggested is 1 -5 degrees.

These data then can be processed further by use of “Image line profile” tool. This tool for now has it’s own “mindset” and does not properly update always. The dependencies are quite complex. If it does not update, close the tool and reopen…



The “SquareMap of Intensity vs pixel” graph on the top right above shows the intensity in linear/log (checkbox left top corner) as function of pixel (bottom axis) and azimuthal angle (left axis). The lineout plot at the right bottom shows the intensity from this plot (note, the log/lin scaling in the image translates here!) as function of pixels/q/d/2 theta. Note, that this produces “natural” binning with every step in pixel is assigned single q/d/2theta position.
Note, the controls:
Number of sectors
Width of each sector - it is possible to have width such, that bins overlap, touch or do not touch… Default here is to have them touching.
Start Angle (0 = right horizontally from beam center)
End angle (wrt to start angle, most likely 360 degrees, or 180 degrees for only top half).
Mask data this tool does not mask, unless selected here…
Note, that by selecting larger width here, one can get very good and reliable sector average and manually move this average through the different azimuthal angles. Very useful, when hunting for particular azimuthal orientation…
Use RAW data if selected unprocessed image is used.
Use Processed data if selected processed image is used, available ONLY if the last image was loaded using one of the “**Convert…**” buttons, unavailable if the last image was loaded using “**Ave & display sel. files(s)**. If the data were loaded using “**Ave & display…**” button, processed data do not exist.

Controls on Lineout tool:
Orientation of line profile (Horizontal/vertical)

X axis linear/log scale
Use: pixels/q/d/2 theta
Width and position
Save lineout – this saves “qrs” data in SAS folder in current Igor experiment. Suggested folder/data name is offered through dialog and user can modify as needed. Note, that errors are simple sqrt(intensity) – another words, these errors are not very useful.

LineProf

This tool calculates Intensity profile along curve on the detector. It uses different method than **Sectors** tool. Therefore, there are some important differences in how to use this tool…

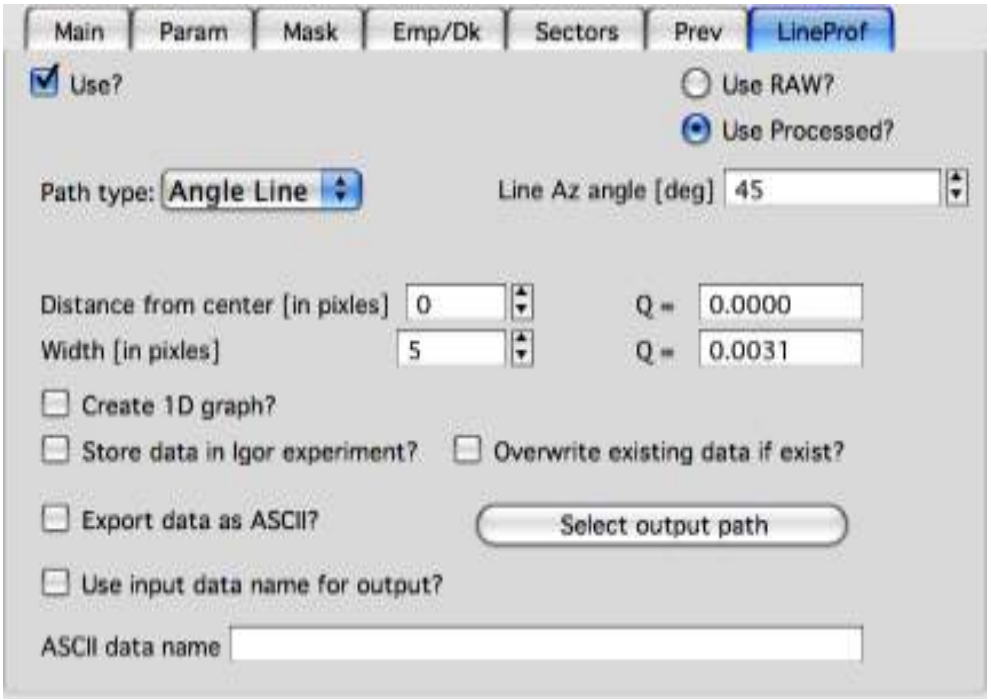
The differences:
"**Sectors**" use inverse lookup method and can be set to create multiple different sectors on one image at once. Since this tool caches the lookup tables, it is slower first time, but much faster on subsequent images. This tool can be used ONLY by setting the data reduction parameters and then using buttons "**Convert…**". You cannot manually evaluate any sector and no preview is provided. This tool causes high memory sizes of the Igor experiments with Nika package – the lookup tables are large. But it is fast for what it does.
And you can setup multiple sectors to be evaluated at once.

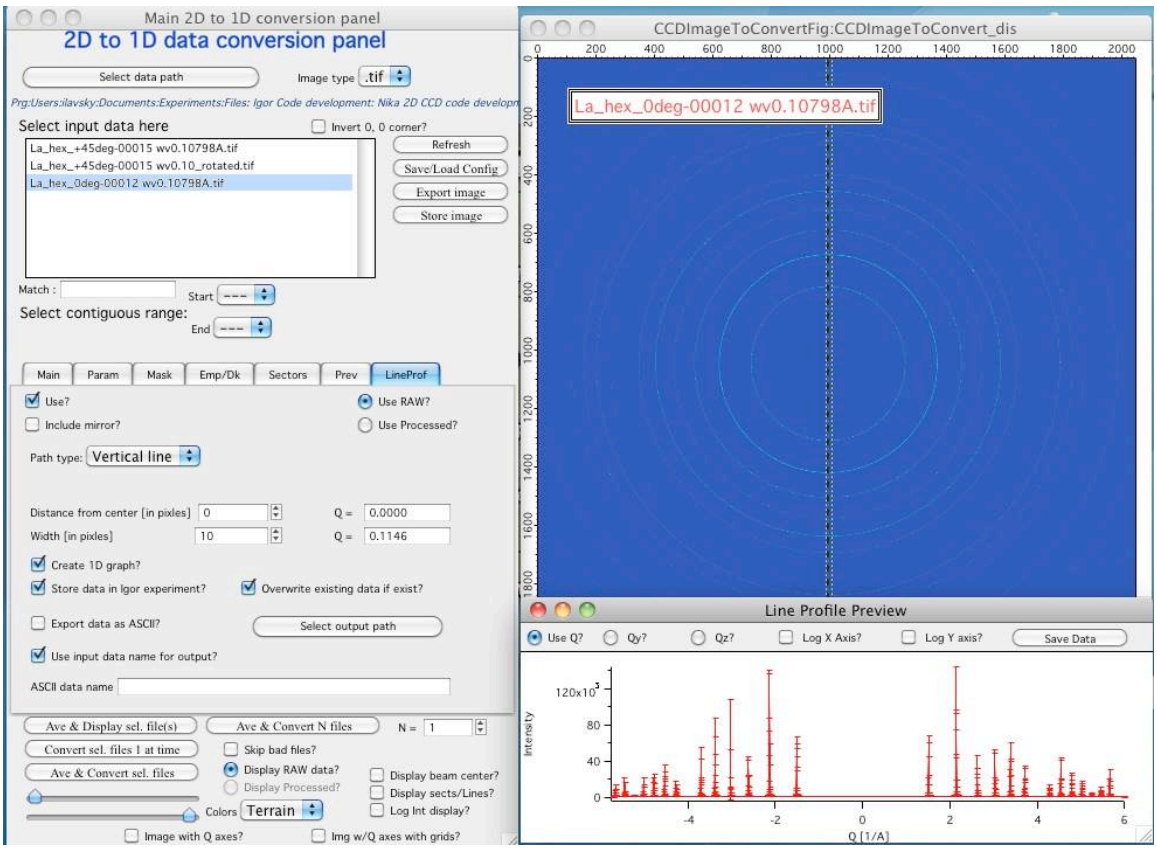
"**LineProf**" uses built in Igor Line Profile tool. It can be set ONLY to process one line profile at a time. This tool does not cache anything, so it takes the same time to process for each image. However, it is relative fast and can be used manually on Converted image. So, there are two methods to use it:

a. Set one line profile parameters, choose how to save data and push one of buttons "**Convert..**"
b. Do not set any conversion parameters, but use one of the buttons "**Convert..**", set the **LineProf** tool to use Processed data and then set parameters for the

You can only set one line profile at a time, unless you manually create multiple profiles on each converted image.

Controls:





New controls here:

"**Use?**" – switches on this tool.

"**Use Raw**?" – and "**Use Processed**?" – choices which image the tool will be used on. User Processed is not available if the last data set was loaded using "**Ave & Display..**" button (no Processed data are created in this case). NOTE: if you hit any button "**Convert..**" and this tool is enabled, it is set to "**Use Processed**" automatically.

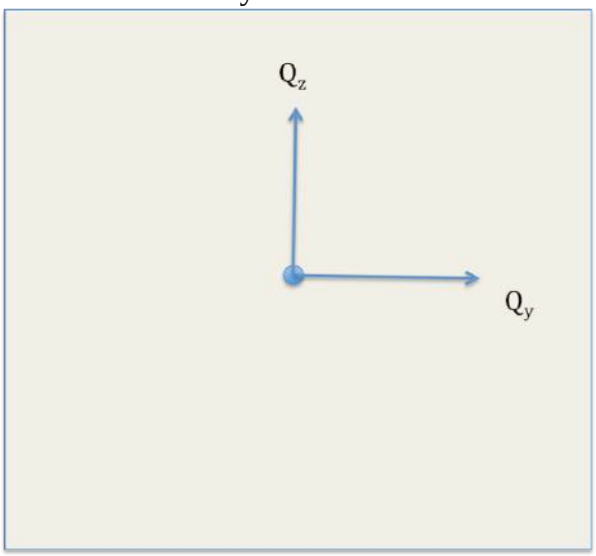
"**Distance from Center [in pixels]**" – user control to move the object to specific q . The q where the data will be calculated is displayed next to this control and is the appropriate q (q_y or q_z) for give shape. See Ellipse definition for specific there. NOTE: you must control the pixel position. Positive direction is to the right of the beam center (horizontally) or up from the beam center (vertically). Lines are drawn to help user figure this out.

"**Width [in pixels]**" – width of the profile (minimum used one is 1 even if 0 is set by user) in pixels. This is the control to use to change how wide stripe is averaged. Next to it is control which shows this in q units. NOTE: the q width is calculated simply by subtracting Q values for the sides of the stripe. Intensity is averaged at each point perpendicularly to the direction of the line (curve). If more than 1 pixel is used for averaging, standard deviation of average is provided as error, if only 1 pixel is used, square root is used (which may be seriously WRONG)… You were warned.

This tool calculate intensity, intensity uncertainty and q, q_y, and q_z values. If one of GI profiles is used, it will calculate q, q_y, q_z, and q_x values. See below.

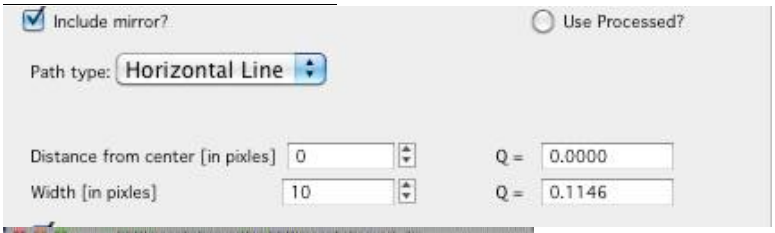
IMPORTANT:

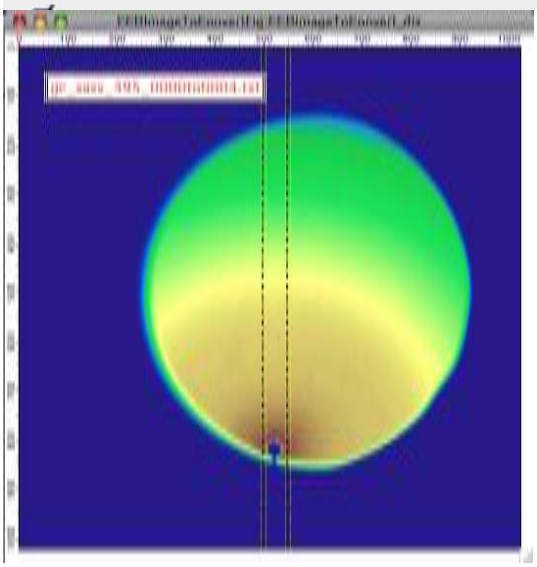
Of course, GISAXS community had to adopt different definition of Qx, Qy,a nd Qz than I did years ago, and therefore, this tool uses somehow different definitions than rest of Nika. So the horizontal direction (x-direction for Nika) is the Qy direction. Vertical direction on the detector is “y” direction for Nika, but it is direction of Qz. Please, keep this in mind… For those adventurous souls, who actually read my code, keep in mind at some point the code switches on your the x-y image coordinates to y-z-(x) GISAXS coordinates… Sorry. No other fix I would know about.



For now these are the available profiles:

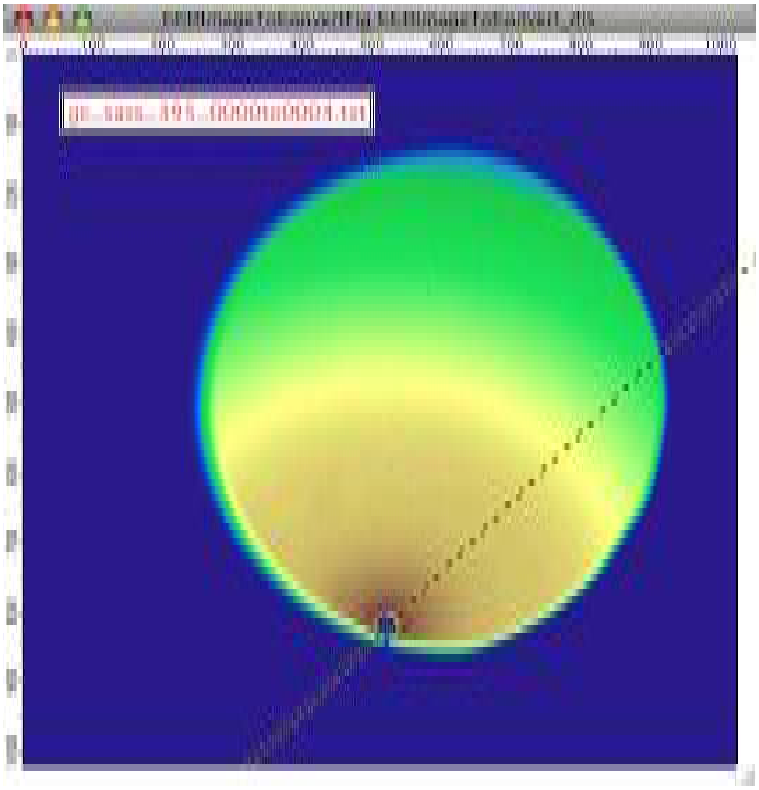
Vertical/Horizontal line:





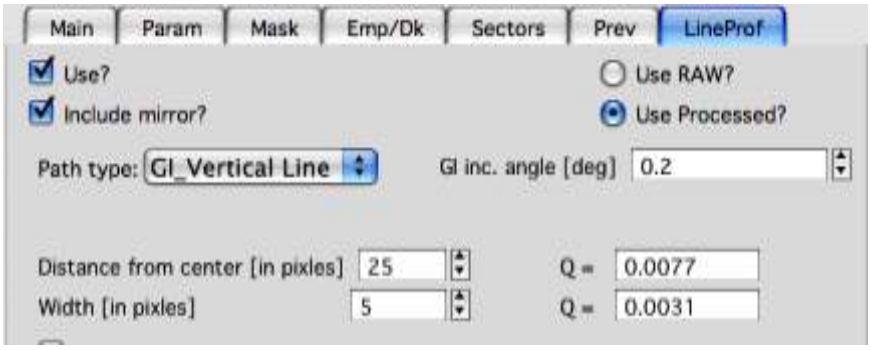
There is one more control available – "**include mirror**" (above the popup). If this is selected, mirror line over the beam center is included in calculations, see above.
This is line profile for transmission geometry.

Angle line:



This is also for transmission geometry.

GI_Vertical line & GI_Horizontal line
These profiles are for Grazing incidence geometry. They need Grazing incidence angle:



Both can include mirror image line across the beam center.

Note, that the position is defined in pixels as before, but the Q values are corrected according to the Grazing incidence geometry corrections, see Gilles Renaud, Remi Lazzari, and Frederic Leroy, Probing surface and interface morphology with GISAXS, Surface Science Reports 64(2009) 255-380, formula (1).

Note: before version 1.68 there was bug in the code for calculation of one of these angles. It hopefully had negligible impact for higher angles, but for small angles the Q calculation was wrong. The fix is, unluckily, complicated – as far as I know, there are two common GISAXS geometries being used. This requires additional user choice here.

Here is the explanation; following pictures are from Lazzari, J. Appl. Cryst. (2002). 35, 406-421 and G. Renaud et al. / Surface Science Reports 64 (2009) 255–380):

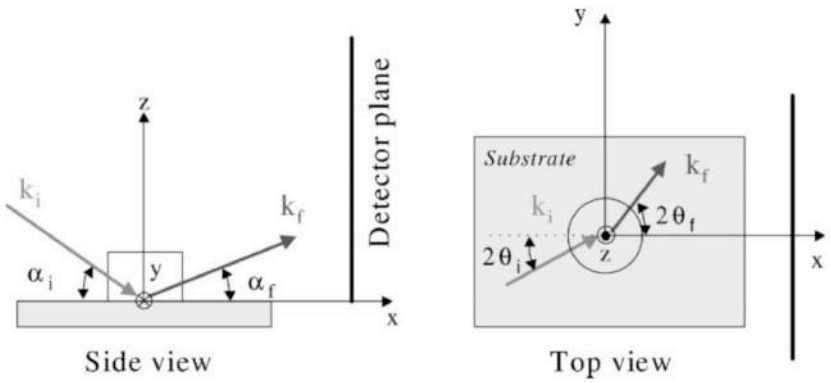
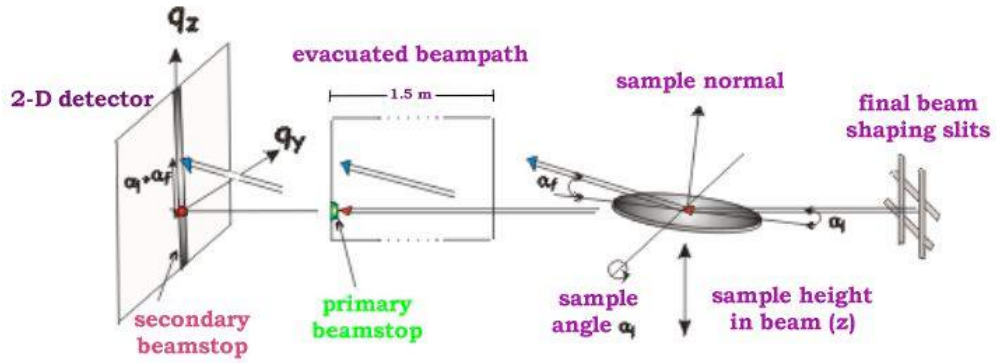


Figure 1
Sketch of the grazing-incidence geometry: an incident wave of wavevector $\mathbf{k}_i$ is scattered in the direction $\mathbf{k}_f$.

Here are the q components calculations based on this geometry. Note, Nika assumes Theta-I = 0.

$$\mathbf{q} = \frac{2\pi}{\lambda}\begin{bmatrix} \cos(\alpha_f)\cos(2\theta_f) - \cos(\alpha_i)\cos(2\theta_i) \\ \cos(\alpha_f)\sin(2\theta_f) - \cos(\alpha_i)\sin(2\theta_i) \\ \sin(\alpha_f) + \sin(\alpha_i) \end{bmatrix}.$$

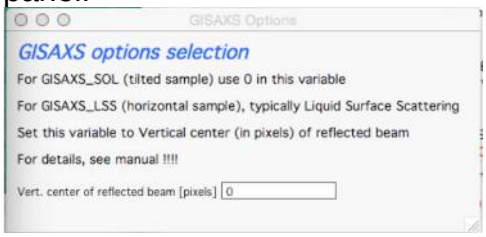
However, another geometry, which is also used, is slightly different:



(Fig2. - http://www.physics.queensu.ca/~saxs/GISAXS.html)

Note the difference here is, that in the first image the sample is horizontal and beam is tilted, as it is commonly used for liquid surface scattering (“GEO_LSS”). For solid samples it may be more convenient to tilt the sample itself and rest of instrument stays fixed (“GEO_SOL”). In my rare encounters with GISAXS technique, this is what I have used.

These two geometries differ in the calculation of alfa-f needed for calculation of q in vertical direction. For GEO_SOL the detector is perpendicular to the original (incoming) beam direction and the alfa-f calculation does not require any more input from user as the calculation is simply the angle of the outgoing triangle – alfa-I as shown in Fig 2 here.
For the GEO_LSS as in Fig 1 the detector is perpendicular to the sample surface, and principally user should provide one more input parameter, as the triangles are not right angle any more. In this case users need to input another value – y position of the reflected beam.
Therefore if user selects GI geometry, from version 1.68 he/she should get new panel:

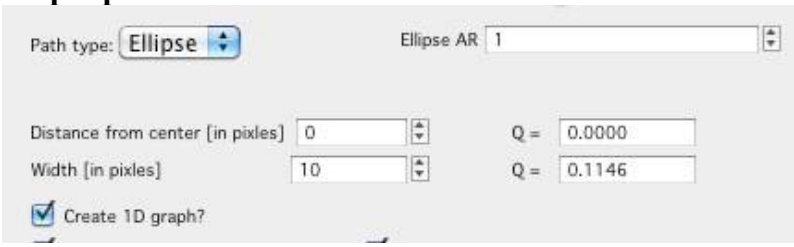


As instructed, for GISAXS_SOL where sample is tilted, just put (or leave) 0 in this field, close the panel and all is OK.
If you are using GISAXS_LSS geometry, you need to read (in pixels) position of the reflected beam and provide here the y coordinate of this beam. Close the panel and all should be set. Nika will use GISAXS_SOL calculation if this value is set to 0 (actually, if it is smaller than 1), and GISAXS_LSS if this value is larger than 0 (actually, >=1).
I do not have chance to test this, so if someone can test this and verify this all works, I would be really grateful.

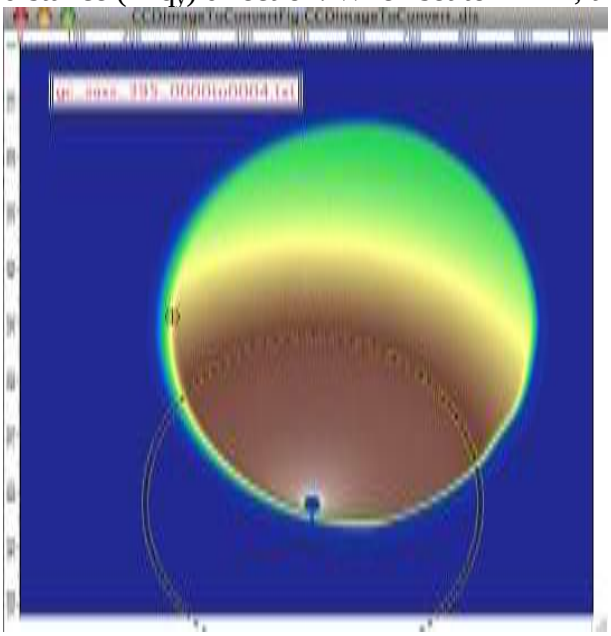
And interestingly, there are instruments, which move their area detectors around much more, and orient them in much more complex way – and Nika has simply no chance to handle those systems. More complex instruments will require dedicated data reduction software.

The bug in this angle calculation was found by one of the users (Thank you!) in version 1.67 of Nika – the correction for alfa-I was missing.

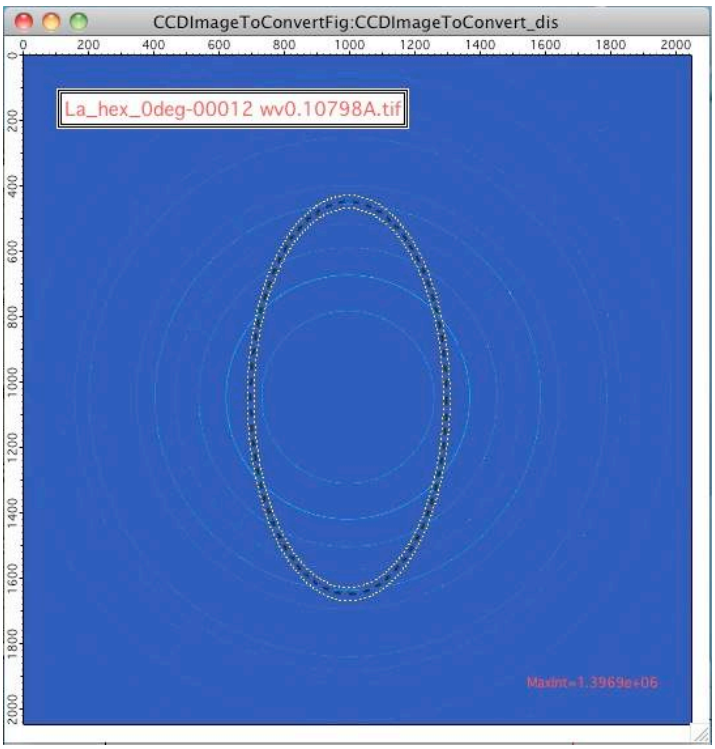
Ellipse profile



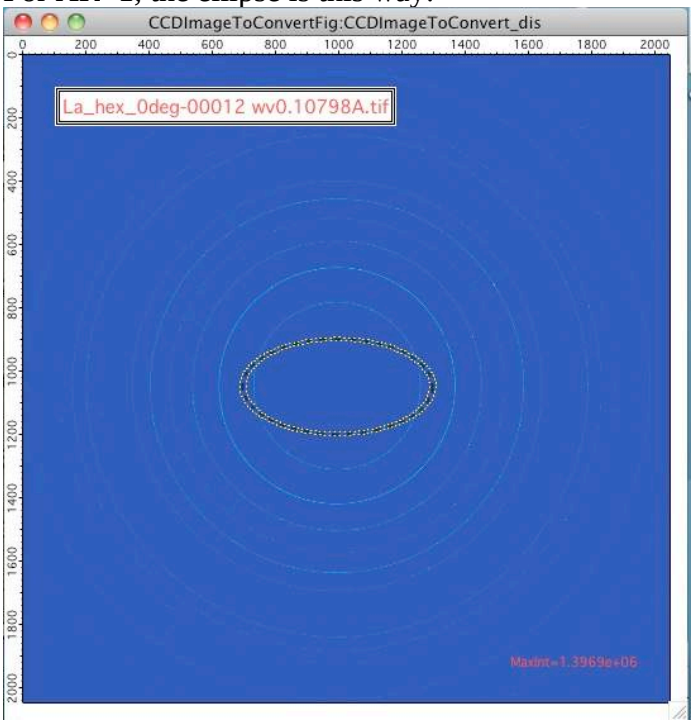
Note, that there is aspect ratio control here and the Distance from center here is horizontal distance (in q_y) direction. When set to AR=1, the ellipse becomes circle.



For AR>1, the ellipse is this way:



For AR<1, the ellipse is this way:



Note, that this tool has one major problem – it is practically impossible to display the data in any sensible way. Neither q, q_z, or q_y makes any sense here. In some way one needs to get angle of the intensity position. At this moment I do not produce such data within Nika. User can produce them by himself (the step is 0.25 degree, starting from 0 degrees

azimuthal angle on the detector[note: I hope, I got turned around so many times, that this requires some data to test on]).
The other option is to use q_y and q_z to generate this angle. **If anyone will ever use this tool, please, contact me and tell me, how you want to use it and I will modify the tool to suit needs of users.**

Finally : More shapes…. I can imagine broadening capabilities of this tool with other shapes. If you have such need, talk with me and I'll add line profile shape for your needs.

Controls for saving data are the same (really, these are the same controls, showing on second screen also) as in the **Sectors** tab:

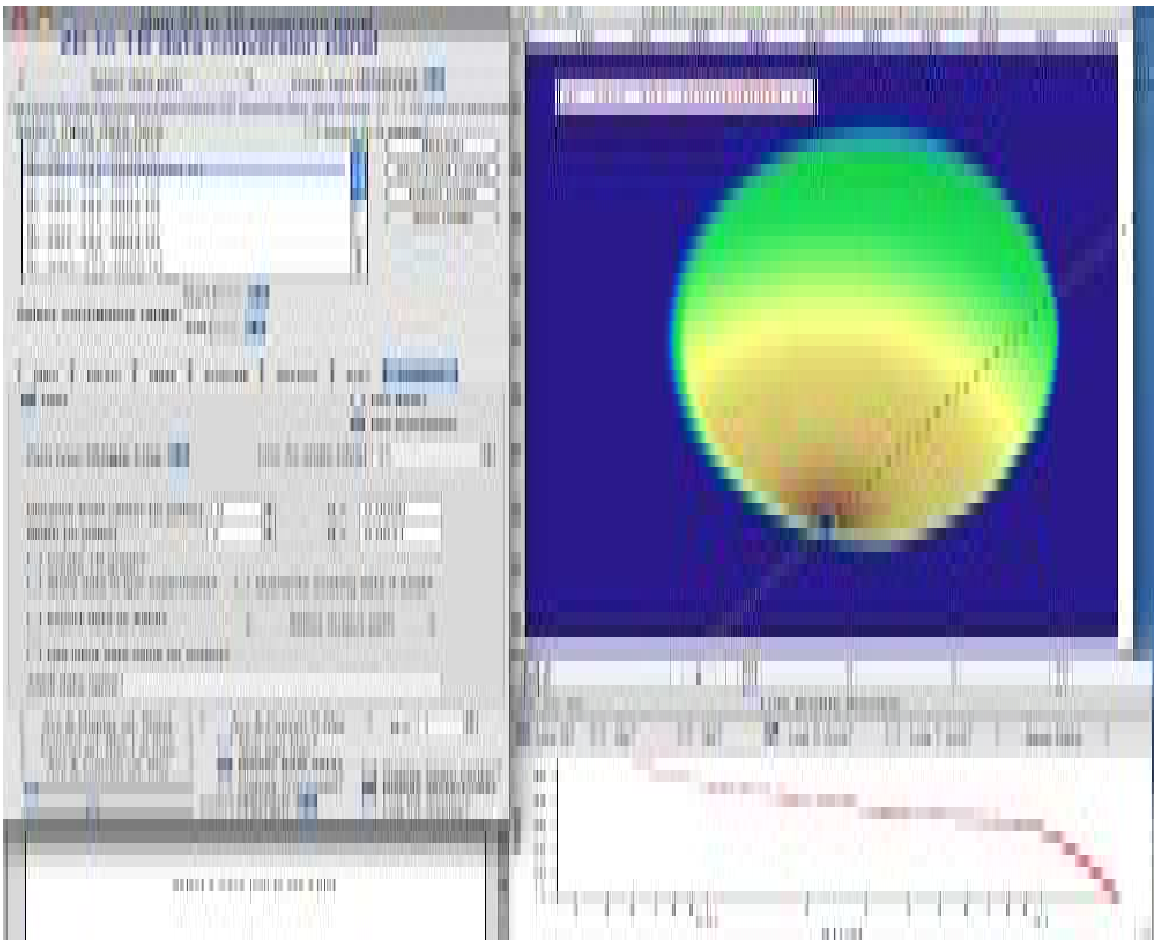
Create 1D graph – if checked, 1d graph with output is created (if necessary) and data added. Note, the graph may be crowded very fast, since data are added, and added…

Store data in Igor experiment – keep data (as qrs triplets) in current Igor experiment. Overwrite existing data if exist – if data with the same name exist, overwrite without asking. Otherwise, you will be asked.

Export data – export ASCII data
Select output path – select where data are to be placed.

Use input data name for output – automatically name 1D data (with sector information added as DataName_Angle_width) by input data name.

ASCII data name – if the above is not selected, this is place to put name for output file. Note, if there is nothing available for the code as sample name, it will ask for some…



Note, that the LineProf tool uses another “graph” window (“Line Profile Preview”) under the main image. This window contains some controls that are very useful.

The data are automatically updated as the parameters for the profile are changed. This gives user live update (but can take time, if it takes too much time for anyone, let me know and I’ll add controls to avoid the updates “live”).

User can display the data as function of q, q_y or q_z and on lin-lin, log-lin, lin-log and log-log scales. Note, that negative values cannot be displayed on log scale, so since q values for lower part of detector (below beam center) are defined as negative, you may not see them if you choose log scale. Also the q values look sometimes really weird, but generally they should be correct. If there are any issues with definitions of negative directions, let me know.

User can also save the data displayed in this window, which enables user to create multiple line profiles from existing image – this is manual method. NOTE that save parameters are taken from the setting of the controls for this purpose in the tab in the main panel (“Create 1D graph”, “Store data in Igor experiment”…). If you choose “Overwrite existing data” and do not change the name, you may get in troubles.

When data are being saved some cryptic description to indicate what profile was used and which *q* was used will be attached to the name used. More full description is attached to wave note.

For example for GI_Vertical line in my test case, this was the name:
gc_saxs_395__GI_VLp_0.0077
"gc_saxs_395_"…. Part of the name of used image
GI_VLp_.... GI_Vertical Line
0.0077 …. q_y value at which the data were calculated.

Exported data are Int, error, Q, qx, qy, qz columns with header and column names
Saved data in Igor are

r_gc_saxs_395__GI_VLp_0.0077	intensity
q_gc_saxs_395__GI_VLp_0.0077	q
s_gc_saxs_395__GI_VLp_0.0077	error
qy_gc_saxs_395__GI_VLp_0.0077	qy
qz_gc_saxs_395__GI_VLp_0.0077	qz
qx_gc_saxs_395__GI_VLp_0.0077	qx (generated ONLY if GI… profile is used)

Note: next release of Irena package will have capabilities to use not only qrs data , but also $q_x$rs, $q_y$rs, and $q_z$rs data.

Polarization correction

Two types are available.

Unpolarized radiation

$$Int_{cor} = \frac{Int}{0.5(1+\cos(2\theta)^2)}$$

This is generally accepted formula.

Linearly polarized radiation

This is polarization correction for linearly polarized radiation, such as produced by double-crystal monochromators on synchrotrons.

There are two polarization orientations, sigma (linear part) and pi. Most synchrotrons will be linearly sigma polarized, with sigma fraction may be 0.99 or so. Depending on the way the detector is read, the sigma polarization plane may be horizontal or vertical. The panel enables setting the sigma polarization plane orientation.

The final formula is:

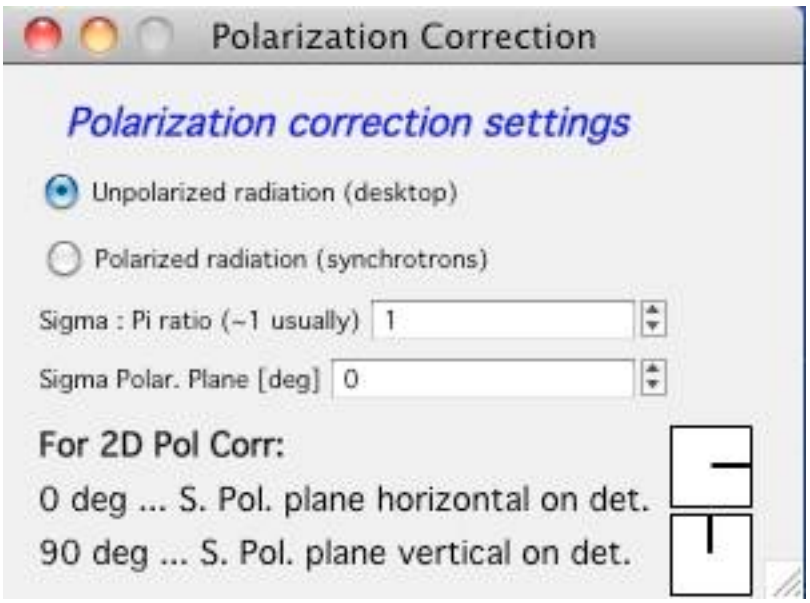
$$Int_{cor} = \frac{Int}{f_\sigma\left(\cos(2\theta)^2\cos(\alpha)^2+\sin(\alpha)^2\right)+(1-f_\sigma)\left(\cos(2\theta)^2\sin(\alpha)^2+\cos(\alpha)^2\right)}$$

where f_s is fraction of sigma polarization, 2q is 2 theta angle, and a is azimuthal angle from the plane of polarization plane.

Implementation

All of the Polarization corrections (from version 1.42) in Nika are applied by scaling the 2D data by the formulas above after all of the corrections (including background and dark current subtraction).

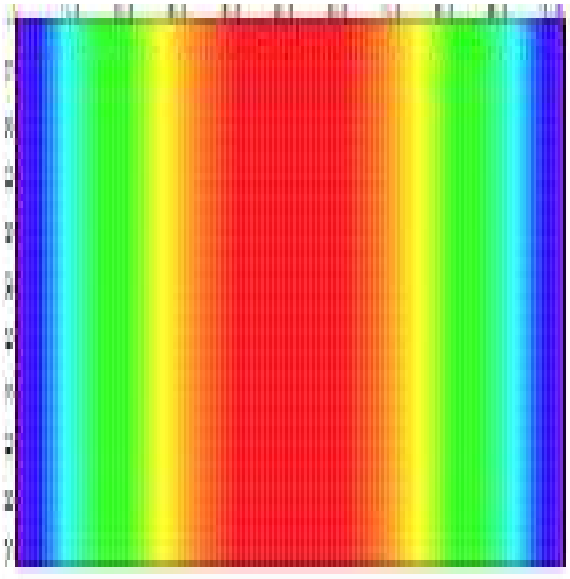
In the following panel which shows after selecting “Polarization correction” on the main panel:



After selecting Polarized radiation you need to make further choice…

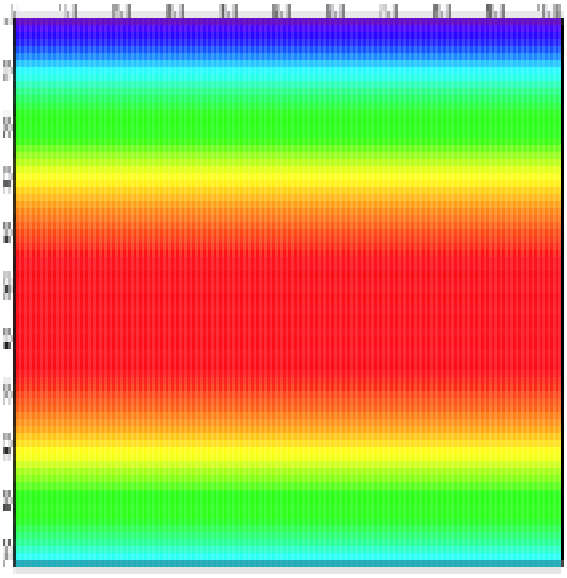
If the Sigma Polarization Plane is 0 degrees, then the detector orientation is such, that the polarization plane is horizontal in the Nika image of the detector. Note that horizontal is Nika's definition of 0 degrees on the detector.

This has nothing to do with the orientation of polarization in real World, this is an orientation between the polarization plane and the way detector is read. In this case the correction looks like this:



with largest correction (increase of intensity) where the color is blue.

For case, when polarization plane is vertical in Igor image (perpendicular to Nika's definition of 0 degrees on detector) , the correction looks like this:



with maximum correction (blue color).

Uncertainties ("Errors")

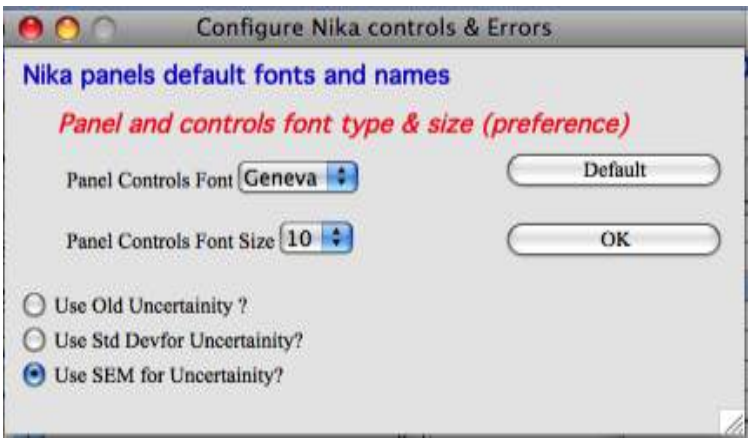
Uncertainty estimate in 2D data reduction is sore point and I have not yet found correct solution for it. As far as I know there is really no good way to get meaningful estimates.

To complicate the matter is, that prior version 1.43 (1.42 and before) there is bug in the uncertainty (error) calculation, which results in overestimate of the values. My intention was to provide standard deviation of the values averaged into the pixel, but simply, I made typo, which resulted in somehow higher values.

Therefore for version 1.43 I provide now three different methods for uncertainity calculations, Standard deviation is default. For compatibility purposes user can choose old (incorrect) version and also standard error of mean – SEM - (standard deviation / sqrt(number of points)).

Please note, that the line profile calculations provide ONLY standard deviation or SEM, since they never used the old method (they use Igor internal method for standard deviation). They default to standard deviation if old method is selected.

The Uncertainty method can be changed in the "Configuration panel" available from menu.

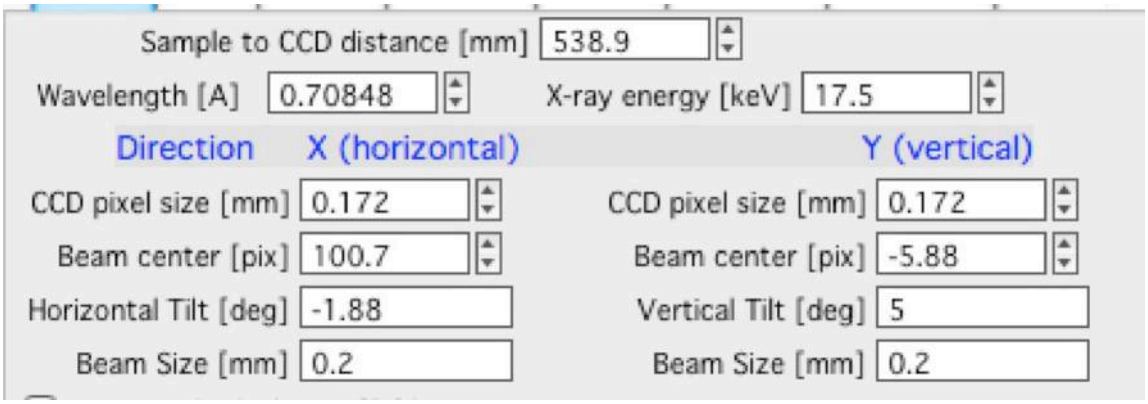


Q-resolution calculations

From Nika version 1.69 the code can estimate q-resolution of the data. This is highly approximate calculation, which can be probably, similar to Uncertainties calculations considered voodoo calculations. I have reviewed some manuscripts which deal with this , such as Barker, J. Appl. Cryst (1995) 28, 105-114. I have looked in some of the codes and realized, that while this is challenge to do for a specific instrument (USAXS code handles this as correctly as anyone probably ever will need), for generic tool this will be challenge. And to some degree, for X-ray instruments this is mostly (not always!) OK as the resolutions are kind of higher than what neutron system need to deal with.
Here is description of what Nika does to calculate q resolution for each point.

1. **Wavelength resolution** is ignored. For regular monochromatic instruments this is reliably ignorable value. For pink beam, well, if you need it I can add it in the future, but I am not sure if anyone needs it (and this would require yet another GUI control value few people would ever use). So if you need it, let me know and we will deal with it then.
2. **Effect of q-binning**. When Nika calculates intensity, it calculates q value for center of each pixel and then generates q binning (linear or logarithmic) – this means, each q-bin has q_{min} and q_{max}. All pixels with q_{center} between q_{min} and q_{max} are counted for each bin. Nika provides this q-width (distance between q_{min} and q_{max}) as q resolution given by nature of averaging.
3. **Effect of pixel size**. Note, that above the q is placed into the bin based on center q value. Of course, this means, that some pixels with center near qmin or qmax contain intensity from q values belonging to other q bins due to finite pixel size. This is q resolution due to pixel size.
4. **Effect of beam size**. Now one needs to realize, that beam has finite size and often is really large. Therefore each pixel will see range of q values (angles) from different places on the beam spot. At the end, this is very similar to pixel size smearing but with beam size values. This is q resolution given by beam size.
5. **Effect of detector pixel bleeding**. This is caused by detectors not being able to separate the intensity in one pixel from the next pixel. This is highly detector

technology dependent and Nika simply ignores it. Luckily, newer generations of detectors (Pilatus) are pretty good in this.



Note, that adding the Beam size q-resolution required adding of controls for the beam size into the main GUI. If beam size is left as 0, the only thing affected is the q-resolution calculation. This is beam size **ON DETECTOR!** not on the sample. If there is focusing, that can cause differences.

OK, so in the table above (and that is not exhaustive table) are some of the sources of the q resolution we need to account for. Nika convolutes together Effect of q-binning, effect of pixel size and effect of beam size. It ignores others.

There are bit more details in how the calculations are handled and in case of real interest, read the code (the function is NI1A_CalculateQresolution in NI1_ConvProc.ipf). It gets bit messy in the way these things get expressed:

1. For “small” q-resolution values caused mainly by pixel size and beam size – and where the q-binning is smallish (or at least comparable) component, the correct is expressing q-resolution as FWHM (full width of half maximum) of assumed Gaussian sensitivity of the q bin across of range of q values. This is what most software assumes. This is what you get always at small qs in Nika.
2. For “large” q widths generated at high-q by log-q binning in Nika (and in USAXS using flyscans etc.) the correct representation is more as rectangular slit smearing effect (similar to slit smeared USAXS instrument itself). This is what you get if you use Nika with log-q binning at higher qs.

Irena Modeling II has been recently updated to handle this type of q-smearing. It is bit mess for number of options ☺

Summary:
Accounting for q-resolution can be helpful for scattering with sharp features (monodispered systems etc…). It may be critical for fitting such systems as I was unable to fit some of these systems without accounting for q-resolution. Keep that in mind when fitting is not going well.

It can also be very useful to look at to decide what is the real q minimum value of any instrument. I have seen cases when device is quoted to have q_{min} – 0.0006 A^{-1} but the q

resolution at that pixel is about 0.002 A^{-1}, which really makes that pixel useless for practical purposes. I think this is more common than we dare to accept…

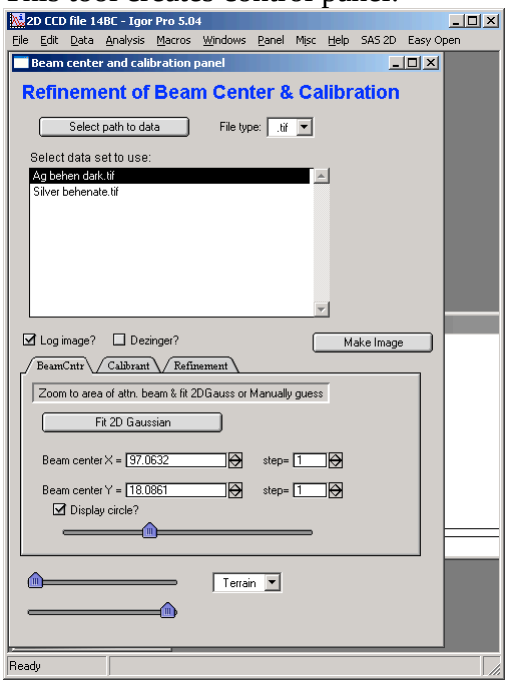
Recently updated Modeling II tool in Irena can handle different types of q-smearing.

Beam Center and Geometry refinement tools

Included tool for finding beam center and refining geometry parameters allows at this time to:

1. Locate beam center when image with attenuated beam is collected by fitting 2D Gaussian profile
2. Locate beam center with help of graphical tools when diffraction lines are available
3. Refine beam center using least square fitting when diffraction lines are available
4. Refine Sample to detector distance, wavelength and beam centers when calibrant image is available (Silver behenate for example for SAXS and CeO standard for diffraction, but user defined is available.
5. Refine tilts for tilted detectors. See notes later. It is NOT that easy.

Main tool is located in the menu under name: “Beam center and geometry cor.”
This tool creates control panel:



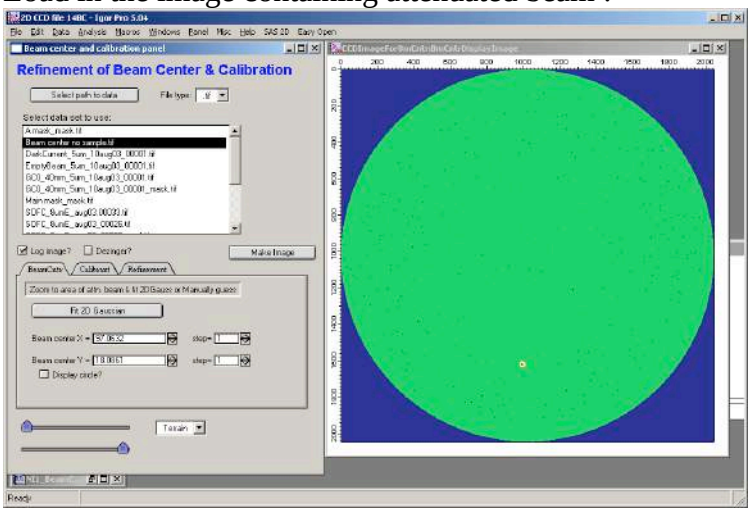
First select path to data, as using the other tools… Select appropriate type of data. Select “dezinger” if needed and number of passes of this process. Check log image if you wish to see log of intensity, but all calculations are done with original intensity… Create image by pushimg “Make Image” button.
If needed (high background samples) you can select “*Subtr. Blank*” checkbox – and need to load Empty measurements (“Blank image”, image without sample but with X-rays on) through the main panel. This is used to subtract that from measured data. It is sometimes needed to have better diffraction peaks.
Other options: “Use mask?” and “Use Geom Corrs?. Use mask – uses mask used in the main panel, so it needs to be first created (suggestion: create mask first before doing

anything else) or loaded using main panel. Geometry corrections to intensity may be important for really high angle scattering, but unlikely…

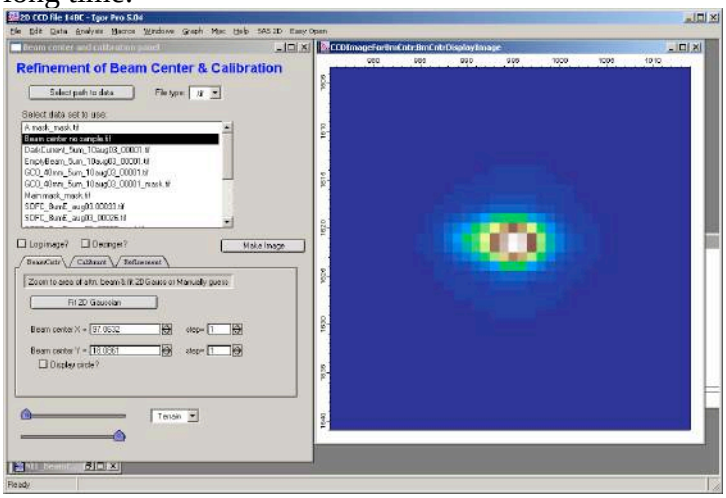
The first tab is for locating (at least roughly) the beam center.

Beam center using attenuated beam

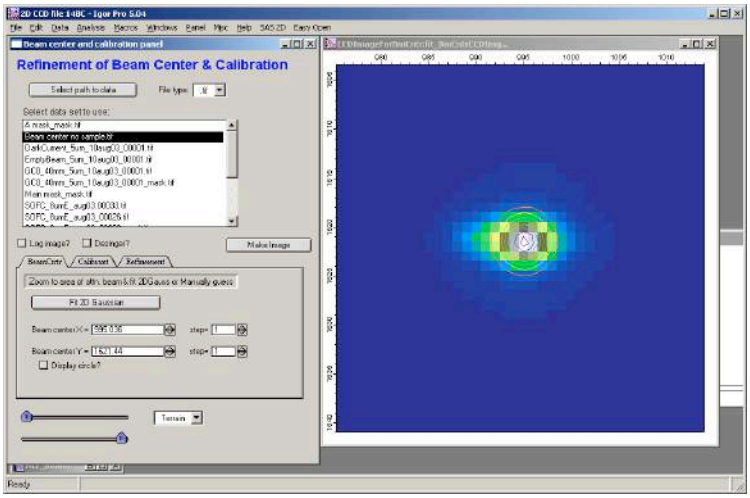
Load in the image containing attenuated beam :



Zoom in the area with the beam using Igor controls (select the area and right-click on Windows, select Expand. Select reasonably small area, fitting over large areas takes a long time.



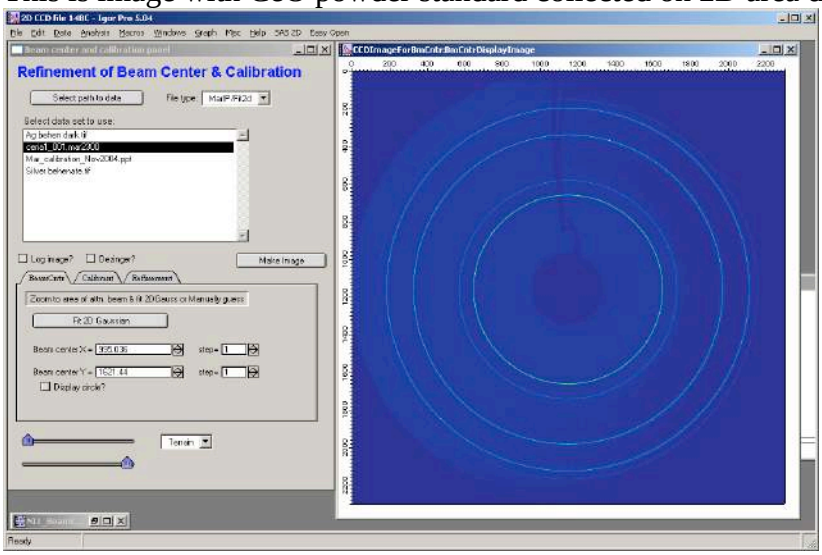
Push “Fit 2D Gaussian” button:



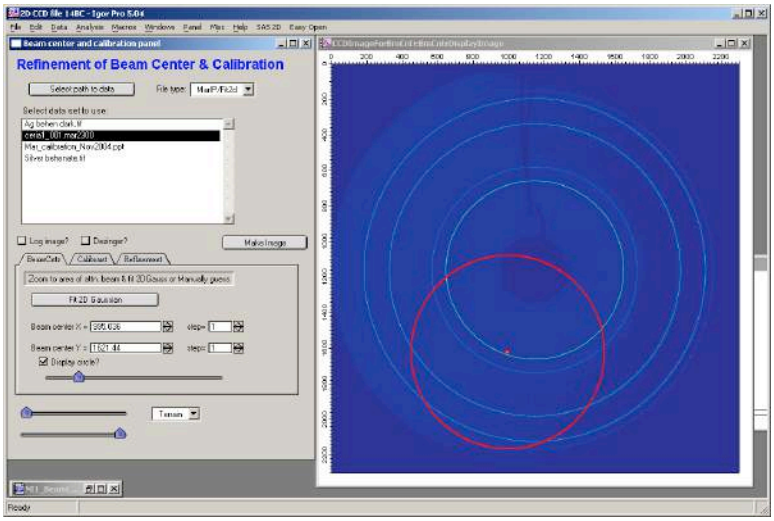
Note, that Contours are appended to the image showing how the Gaussina fit looks like. Results from fitting with beam center values are pushed into right variables.

Beam center using "help circle"

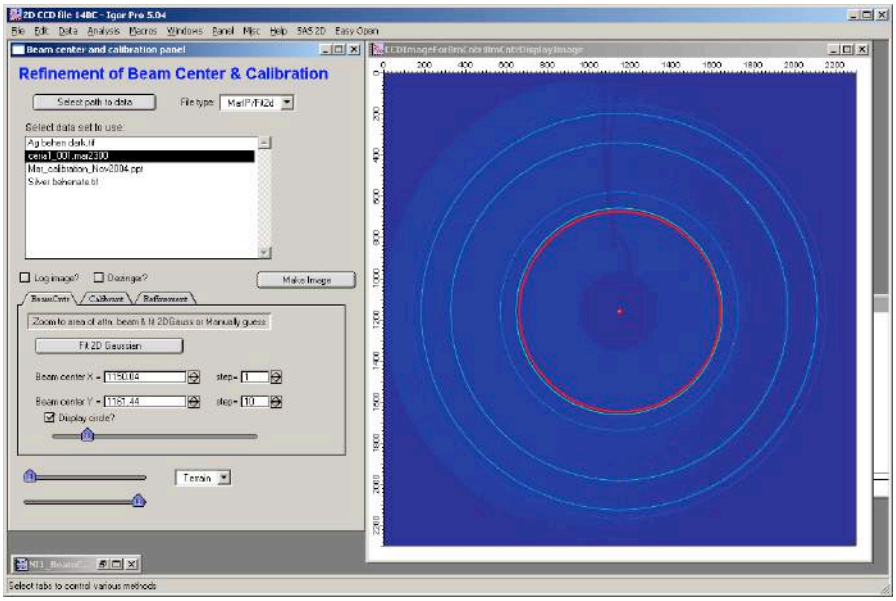
If image with attenuated beam is not available, following method may help to get relatively good estimate for beam center. Needed is image with material which has diffraction rings – this is usually no problem for diffraction, where number of standard exists. For SAXS usual material is silver behenate.
This is image with CeO powder standard collected on 2D area detector:



Check the "Display circle" checkbox and use slider to scale the circle to size close to one of the rings. Then change beam center (set useful step size using the "step" variables) to match the circles as good as possible:
So from this:



Get to this:

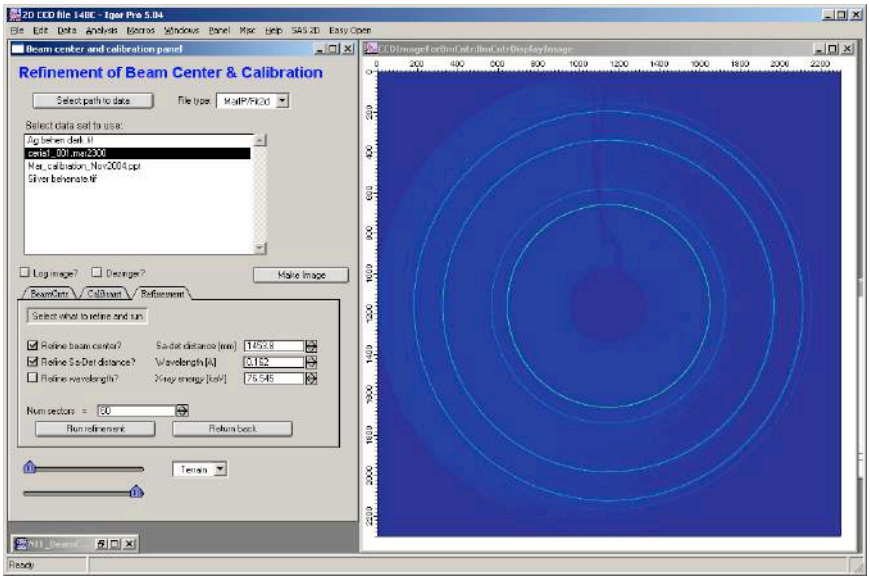


This is already a good estimate of the beam center…

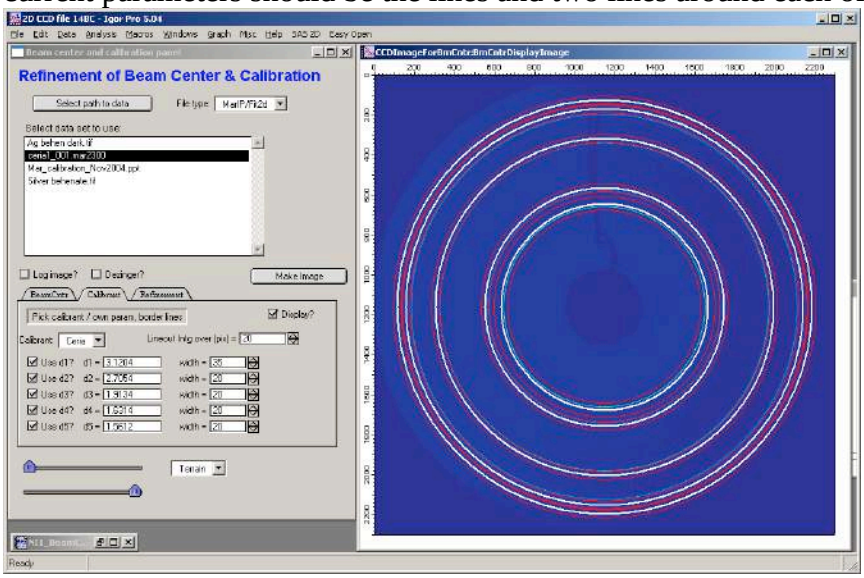
Calibrant & refinement

On the next tab pick the calibrant to use and in tab refinement insert reasonably good estimates of the sample-to-detector distance and wavelength. Pick the predefined calibrant (I have now only CeO and Ag behenate, but can add any number of others). The list of d spacings is filled in… The code can use up to 5 lines for any calibrant material – just overwrite the d spacings on the "Calibrant tab" with own values. User needs to know the d spacing for this material. D spacing cannot be optimized!

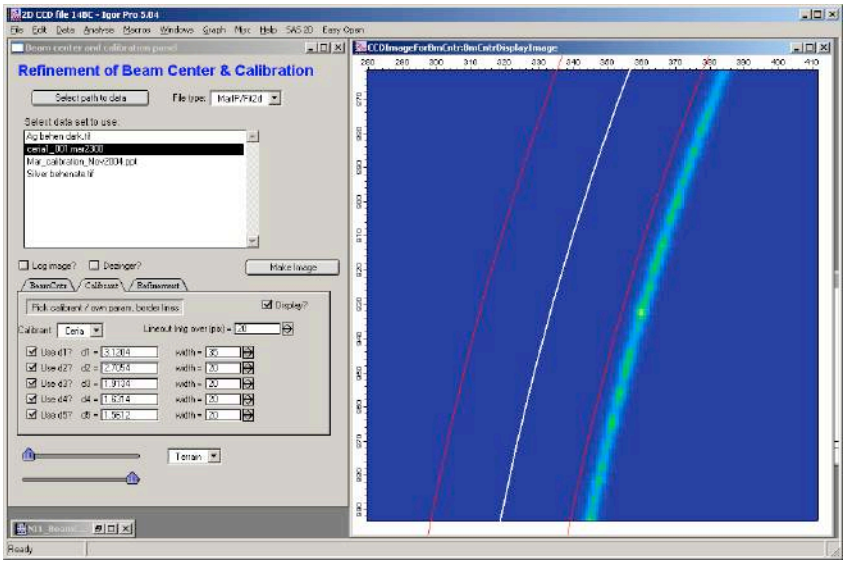
Note, that you have to have also appropriate size of the pixels set in the main panel:



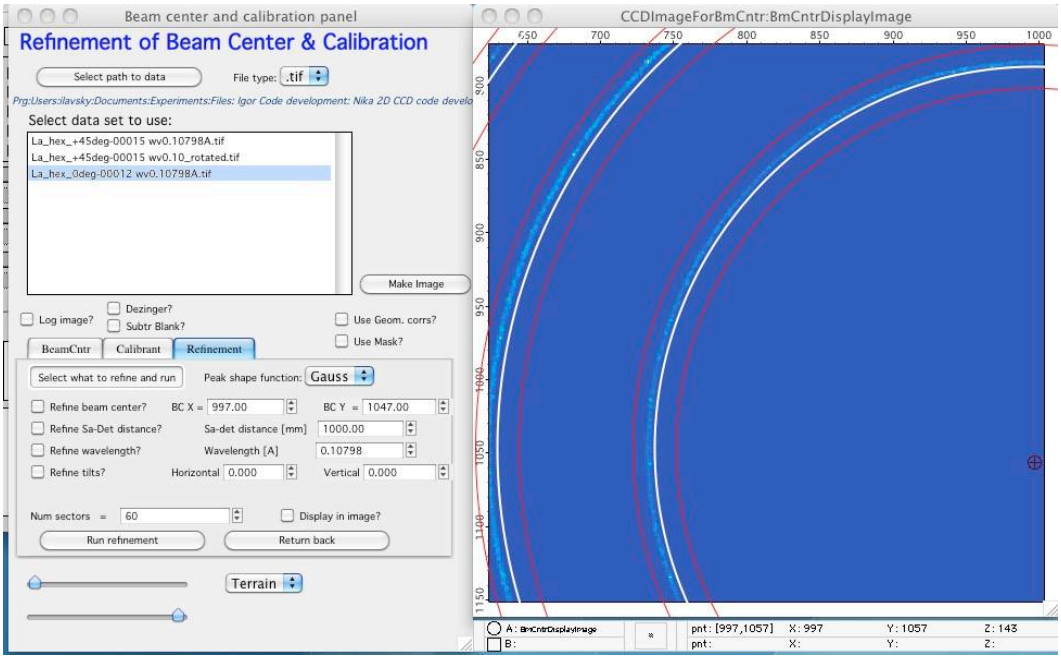
In the tab “Calibrant” now select “Display?” Checkbox. This will add circles where using current parameters should be the lines and two lines around each of this line.



Note detail here:

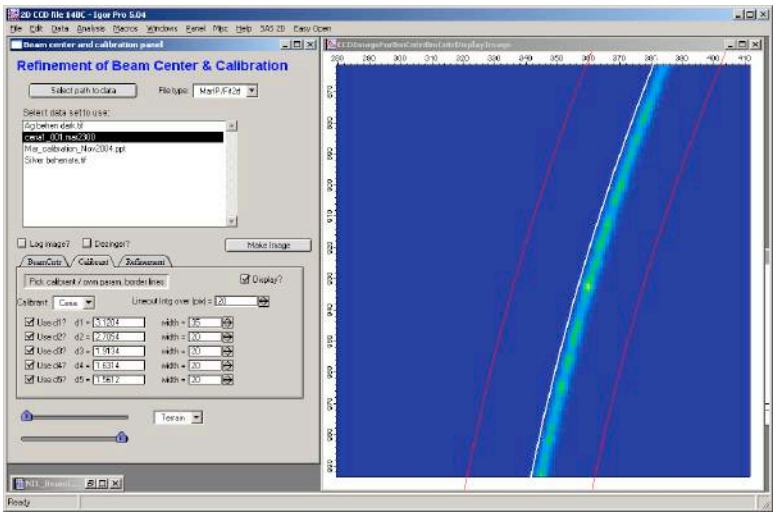


The white line is calculated position of the diffraction from current parameters, greenish fuzzy line below is the diffraction line and red lines indicate the width, which will be used by the code to search for the line positions. In order for the code to work, the diffraction line has to be within the two red lines all way around the circle. It has to be single line within this area – therefore no overlapping lines are possible here…. To do this, change wavelength and sample to detector distance, possibly beam center positions. See here:



If needed make the width between the two red lines wider as necessary – it is line specific, so each diffraction line can have different width. Note, that the peak position is found by fitting Gaussien profile on intensity profile in the radial direction between the two red lines, so they need to contain some flat background around the diffraction line, but now too much.

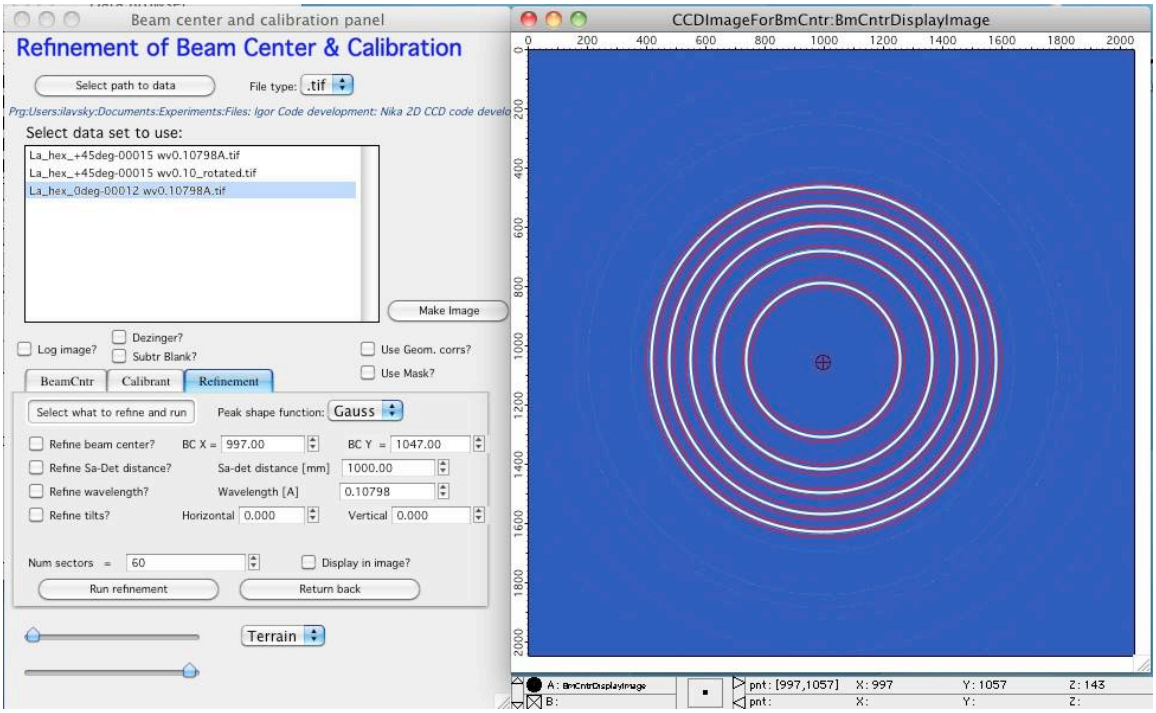
The line profile is taken over width (number of pixels on image) perpendicular to the radial direction as set in “Lineout Intg. Over (pix)” on “Calibrant” tab. This value is ONLY one for all diffraction lines. Depending on quality of the lines this may be narrow or broad. If the lines are broken up, with spots, wider will help, but too wide will reduce precision.



In the “Refinement” tab select which parameters to refine – beam center, Sample-to-detector distance and/or wavelength. Note, that to refine wavelength AND sample-to-detector distance together you need at least 2 diffraction lines.

Select number of “sectors’ to use (see below is set to 60). This how many direction away from beam center are evaluated. For 60 sectors the code analyzes every 12 degrees (360/60=12) lineout in radial direction between the red lines, finds maximum by fitting peak profile and tries to fit to these positions of the diffraction peak.

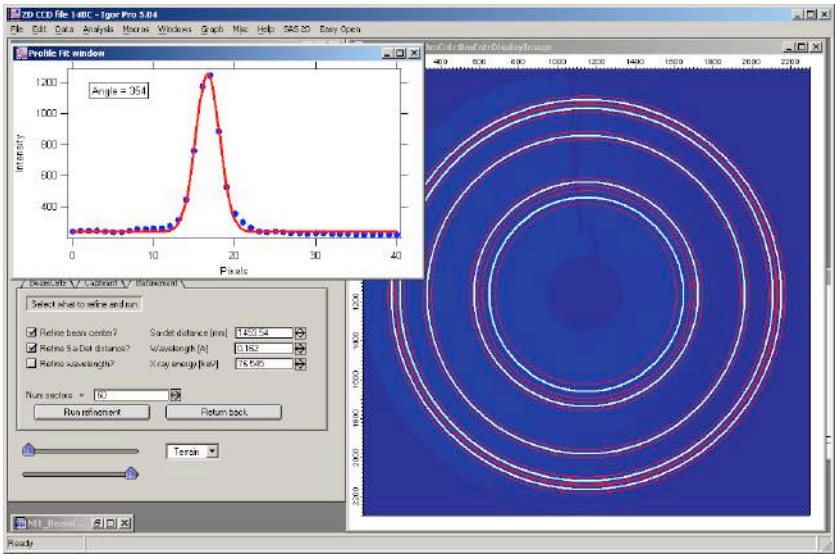
NOTE: Even, if the image covers only small part of the 360 degrees (when beam stop is or beyond the edge of the detector) the analysis is done only every 360/number_of_sectors (in example 360/60=12) degrees. Therefore you may need to increase this number of sectors significantly to make sure you have enough points in which the positions of diffraction ring are analyzed.



This is how many directions for each ring will be evaluated. If the direction falls out of image, it is skipped. Note, too many may take lot of time…
Note: if you select *"Display in image"* the code will show on the image which line is being evaluated at any time. This slows down significantly the fitting as the display part is kind of slow…

Note the other controls:
BC X, BC Y beam center values which can be changed here
Peak shape profile: Guass, Lorenz, and Gauus with sloped background. Most of the time Gauss is fine and most stable. Other shapes are really for cases when Gauss fails.
Tilts… You can change them and fit them here. There is separate chapter later on fitting tilts.

When ready, push "Run refinement" and observe:
As refinement progresses, dotted red line on the image indicates which direction/line are being evaluated and "Profile fit window" graph shows the intensity vs pixel data there and fitted Gaussien profile. Observe and judge quality. If the quality is poor and data are misfit, it is likely that results of refinement will be bad…



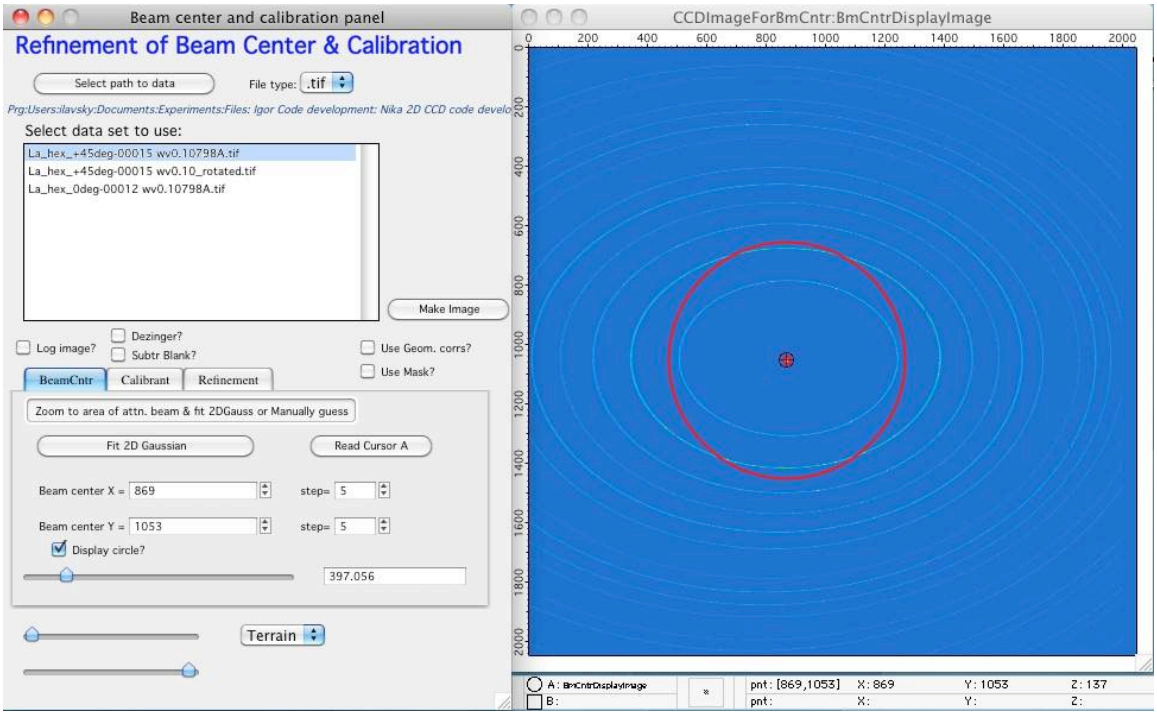
If the refinement at the end fails, you get error message and no change to original parameters is made. If refinement is successful but you still do not like the result, you can recover the previous parameters by pushing button “Return back”.

Otherwise, if successful, the results are pushed into the right variables in the main panel and all is done.

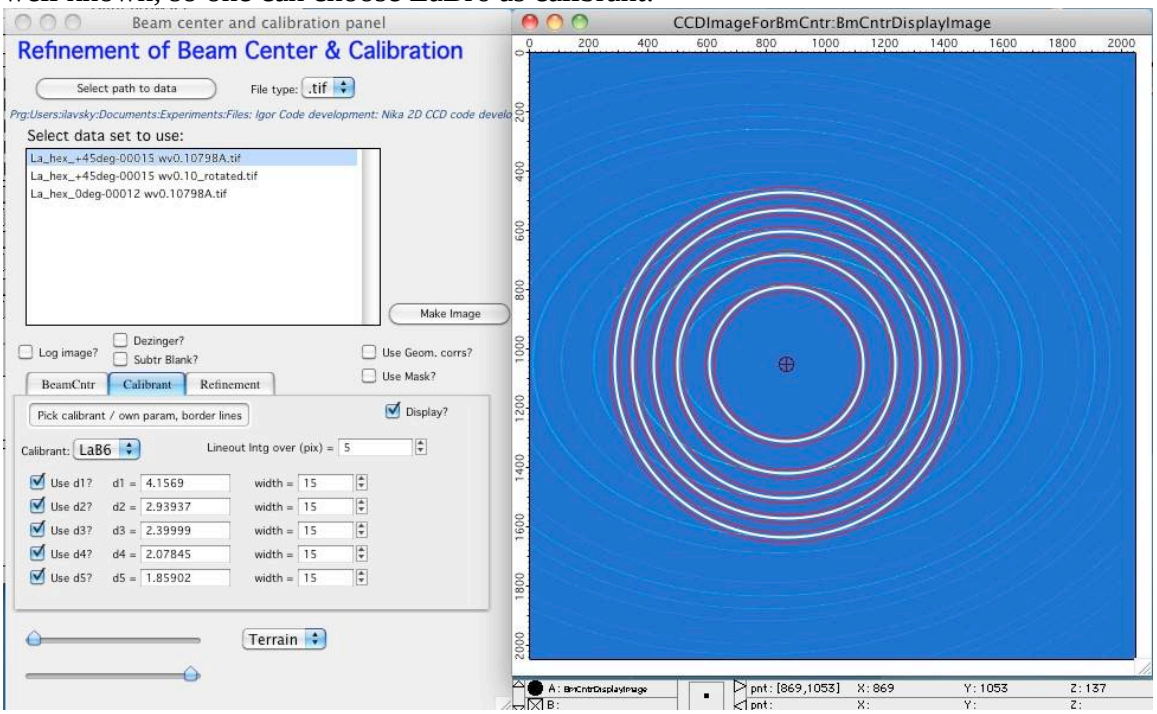
Note, with Silver behenate for SAXS, there is only one line, so the processes is easier. But one cannot refine wavelength AND sample-to-detector distance. Note, the line width for Silver behenate needs to be significantly larger and also it is likely that the “Lineout Intg over “ needs to be larger…

Fitting data with tilts

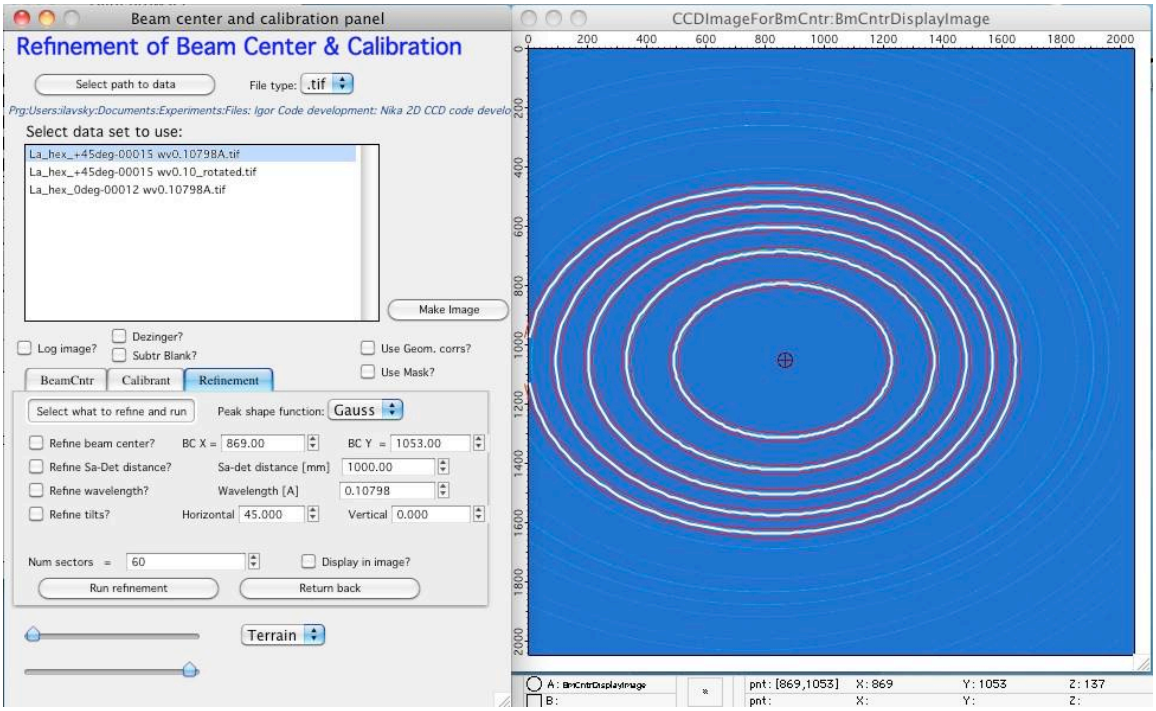
Finally version 1.49 adds good code to fit tilts and deal with them – both in data reduction and in the fitting here. Prior versions (1.48 and before) had slow code which handled small tilts ONLY. Current code, as documented below, handles high tilts quite well and is much faster. Test data I’ll be showing were provided by dr. von Dreele. Many thanks to him.
The following data were collected with about 45 degree tilt in one direction:



Note the deformed diffraction profiles which resemble (but are NOT) ellipses.
Above is the best guess of beam center using the circle. Other parameters are reasonable well known, so one can choose LaBr6 as calibrant:

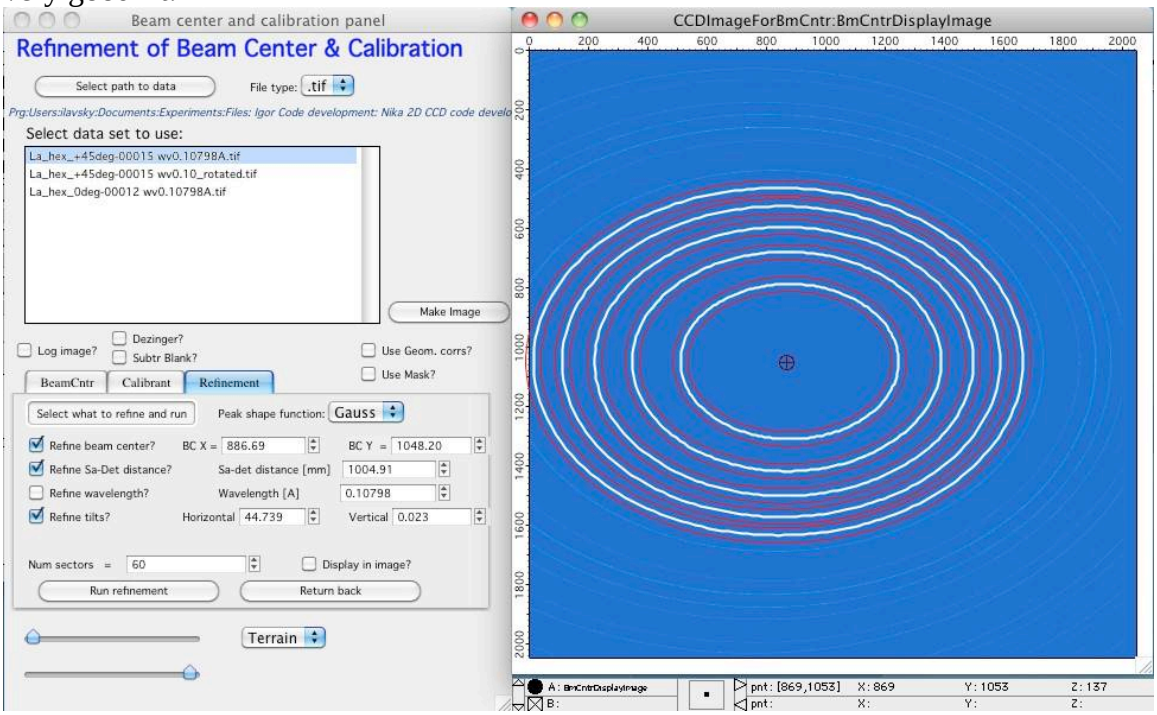


You can see that circles are not a good fit.



However, selecting horizontal tilt of 45 degrees makes this a good guess.

Now we can run refinement for Beam center, Sa-Det distance, and tilts and we should get very good fit:



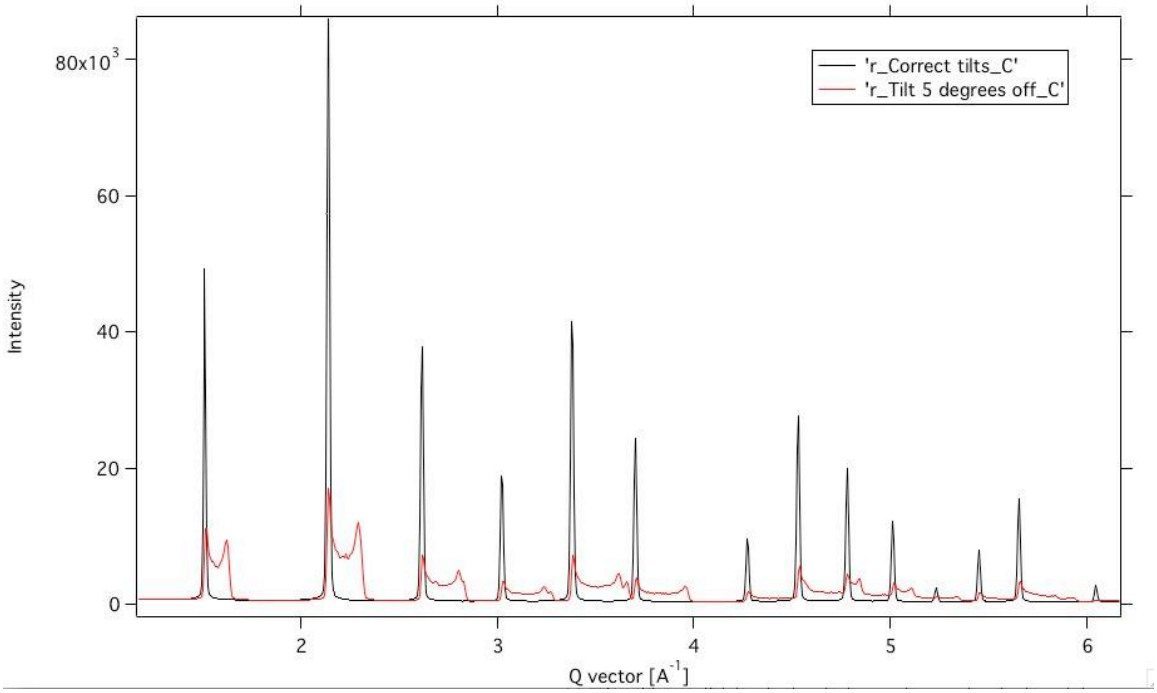
I should note few things:

Make sure the peak fitting does not miss the peak. I try to catch it, but the code is not the most robust. Making the width for each diffraction ring large enough helps a lot. Also, you may want to run the fitting few times. Costs little time and helps often.

Also: Warning – getting tilts requires significant amount of solid angle of data. Basically, you need to see large fraction of the ring to fit tilts. With limited fraction of the diffraction ring my attempts to fit were nearly futile. But you can dial numbers measured by other means in to eyeball the tilts in.

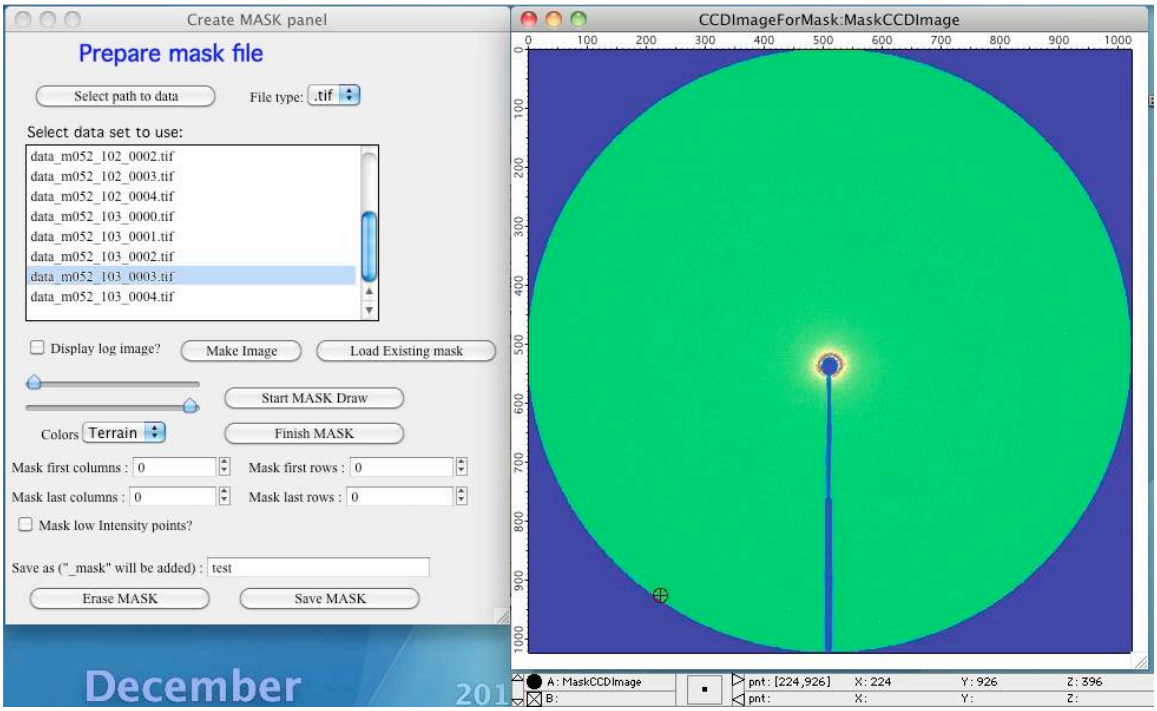
Note that 45 degrees and -45 degrees are NOT the same tilt. There is 90 degrees difference between them, so if you have tilt measured by other means, try using it both positive and negative. Easier to check the effect than try to work out the geometries and convey it here.

Here is example of above data reduced with correctly fitted tilt and with tilt 5 degrees off:



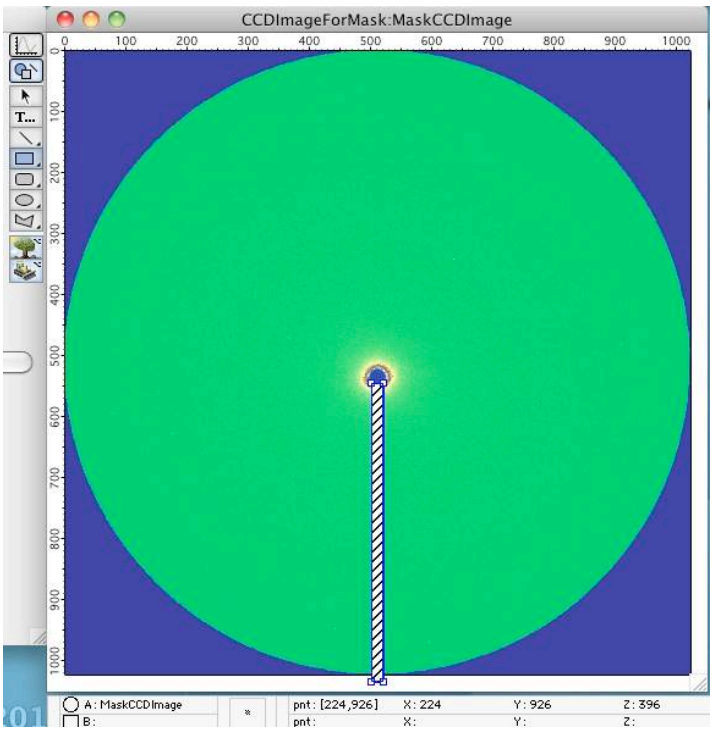
The tilts are important!

Mask

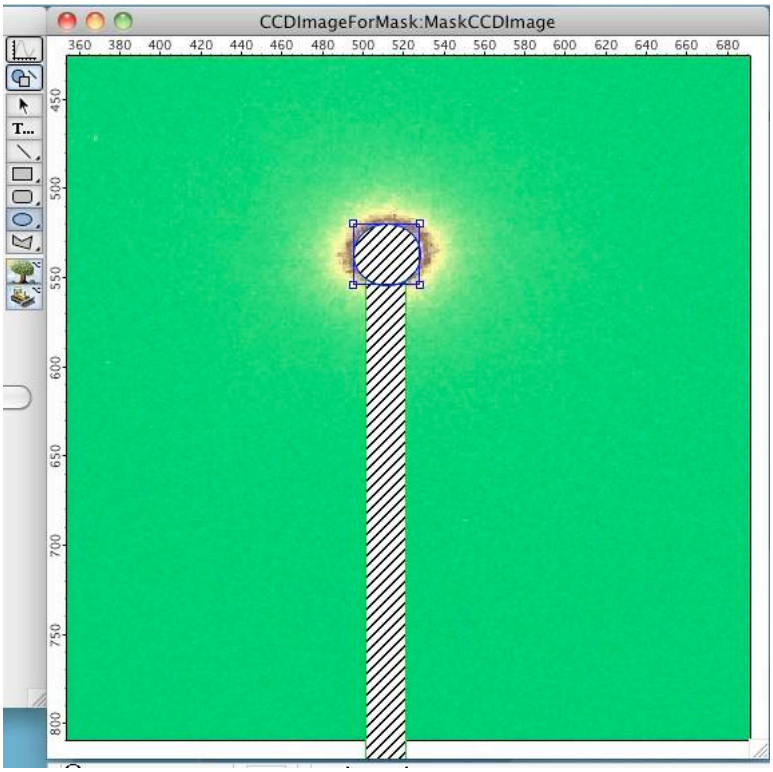


To create mask use tool “Create mask” from either “SAS 2D” or Mask tab on main panel. Select image which you want to use to build the mask from.

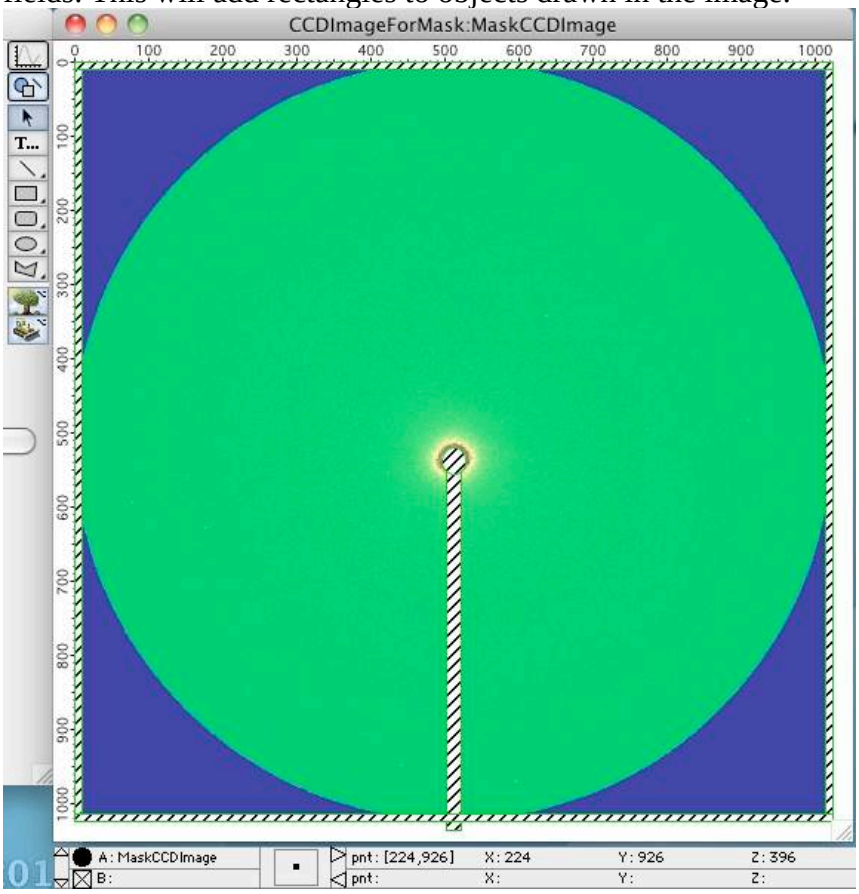
Push “Make image” to create image. Modify by Display log display checkbox, sliders and Colors popup to see details on the image as necessary. Then push “Start MASK draw” to start drawing mask. This brings tools up on the graph.



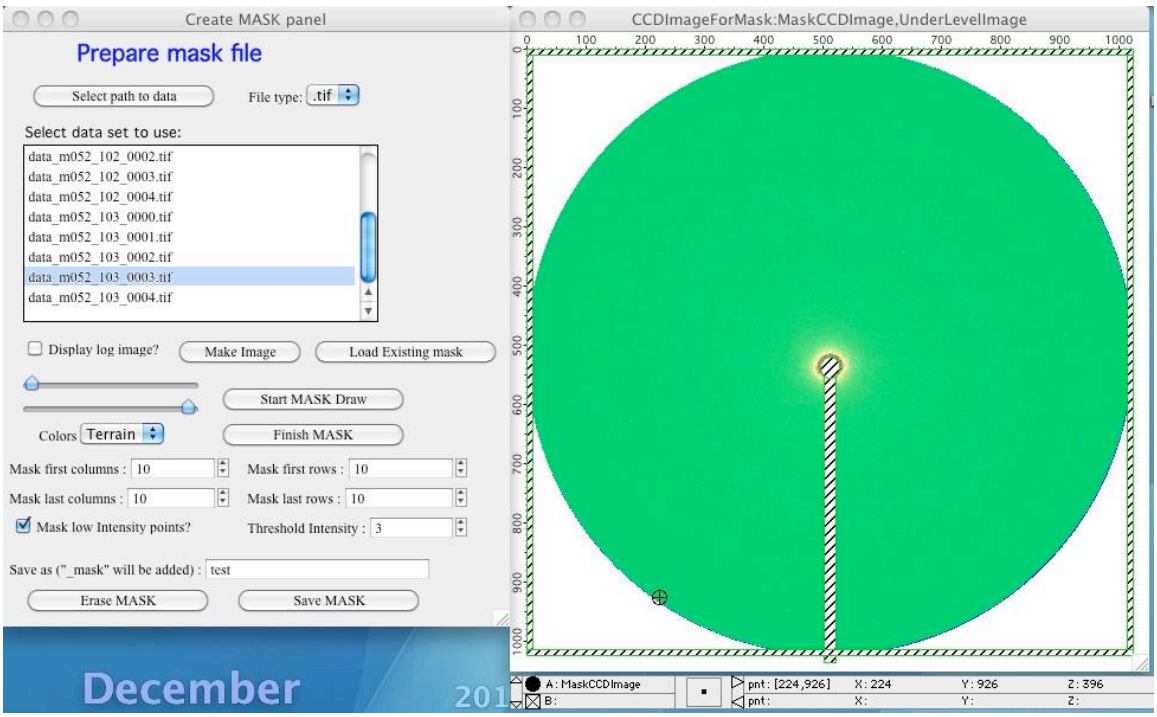
Use Igor tools (rectangle and other tools) to cover area to be masked OFF. You can use zoom in and out to see some specific area of the image and draw with higher precision, but you need to switch between drawing tools and graph tools in the tool on the left hand side of the graph. Note, that on mac sometimes you can have the tools hidden behind the Panel.



If first/last few lines need to be covered, use the “Mask first/last columns/rows” input fields. This will add rectangles to objects drawn in the image:



Next, if the image contains some areas, which have low intensity and contain no information like in this MarCCD image, where the CCD is rectangle, but scintillator material is circle, you can mask off points with “Mask low intensity points” and setting threshold value. NOTE: this value depends if you are using directly image or if you are using log(image). In this case the corners have values set to 0 and so we can mask off anything which is less than may be 3 or so. Image has dark counts/background of may be 100, so it is safe to remove all low intensity points.



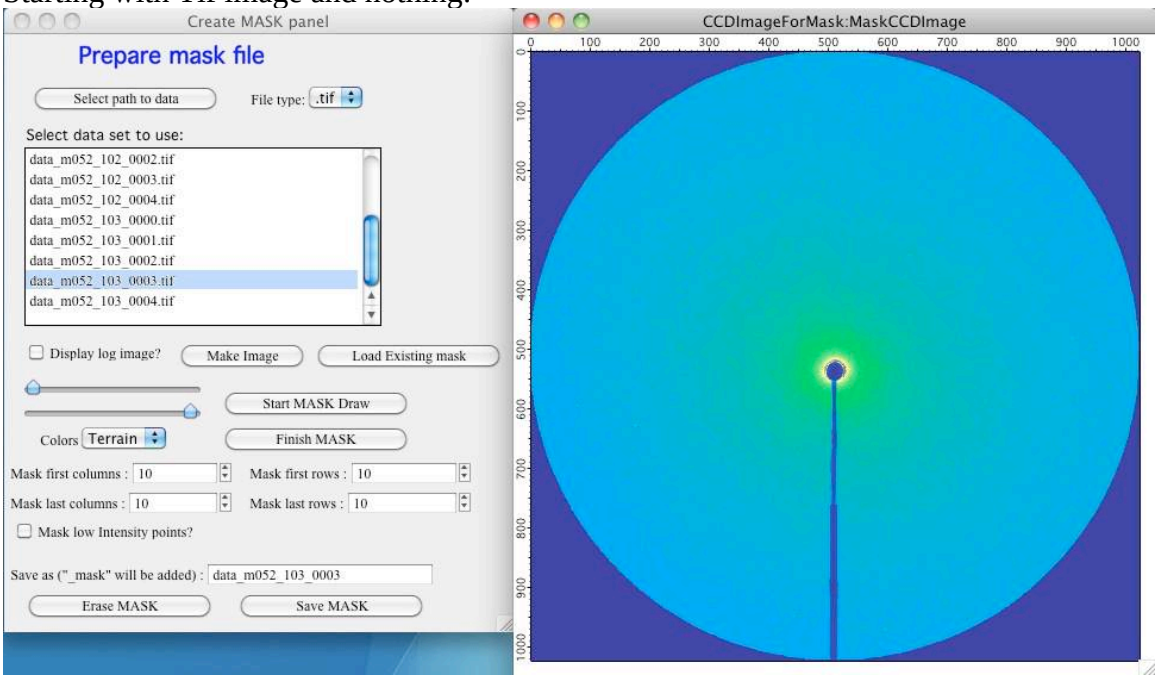
OK, now we should have suitable mask for this case. You can now push button “Finish MASK” and the image will be set to regular mode. To continue drawing you need to push button “Start MASK draw” again. Previous drawing will NOT be removed.

Then we can SAVE the mask, giving it easy to understand name. Code will add _mask.hdf to the name. This mask is now available to the code and also can be loaded later from hdf file as mask to be used.
NOTE: prior version 1.49 Nika used Tiff file for mask storage. It can now load either the tiff file or the hdf file, but it will now save only hdf file, for reasons explained below…

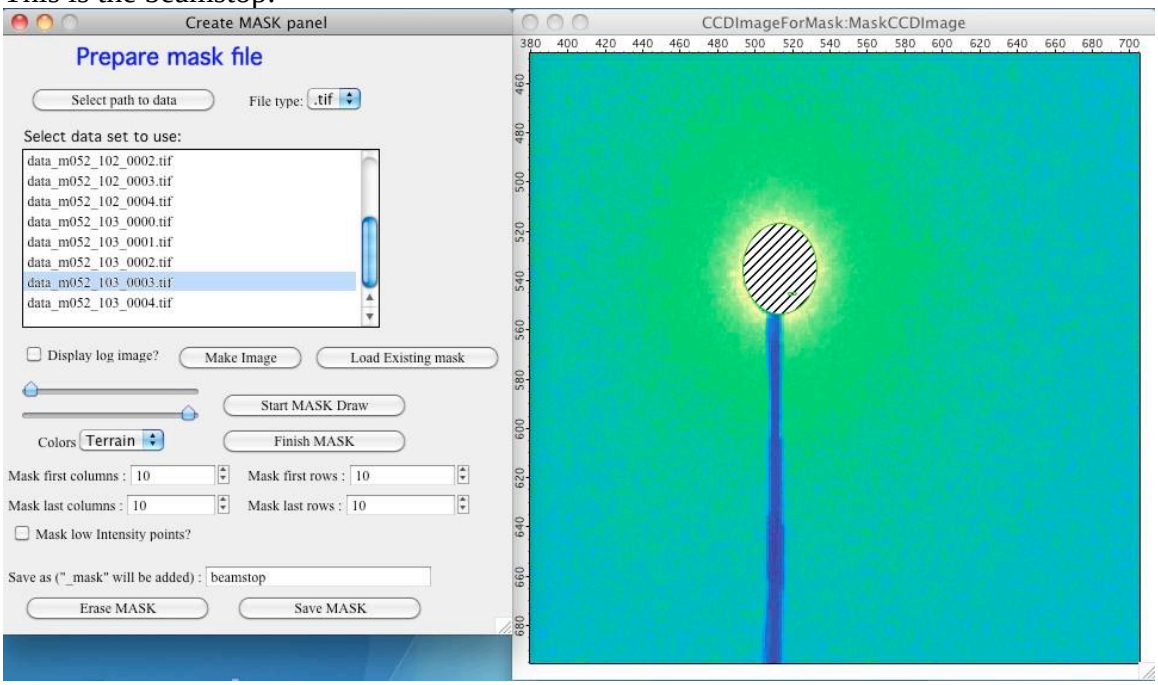
Editing old mask:

I have been asked multiple times to enable editing of existing mask. Using the tiff file this is not possible, so from version 1.49 Nika will use hdf file. This file now stores both image to be used in analysis by Nika as well as recreation macro to be used by “Create Mask” tool. Therefore, one can now store partial masks and combine them later into meaningful combination. Therefore, if there is known mask related to dead or bad pixels on detector, one can store that separately and then use it and always add parts of mask needed for specific setup. Here is any example…

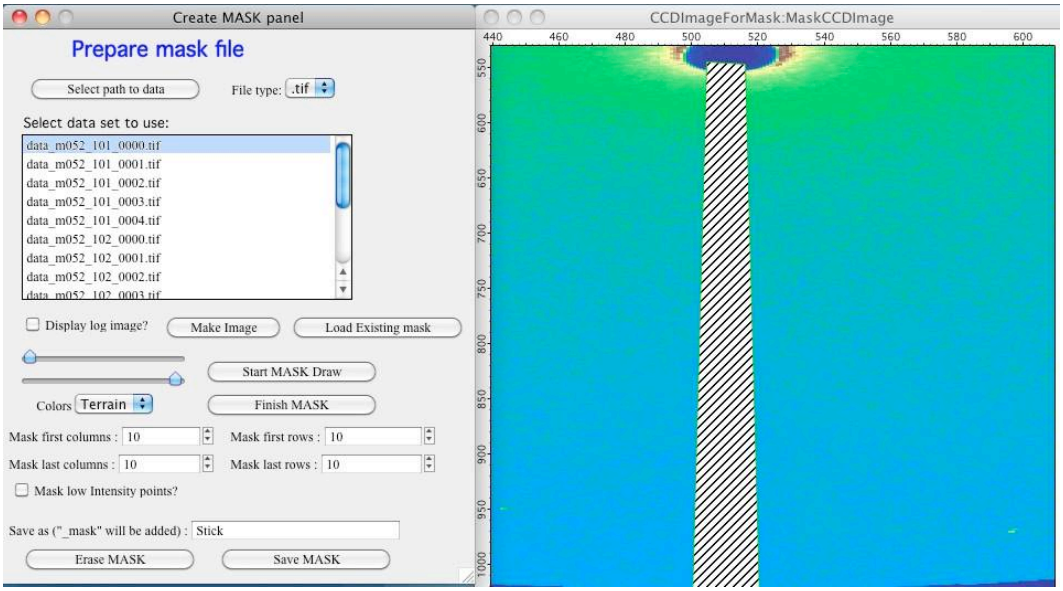
Starting with Tif image and nothing:



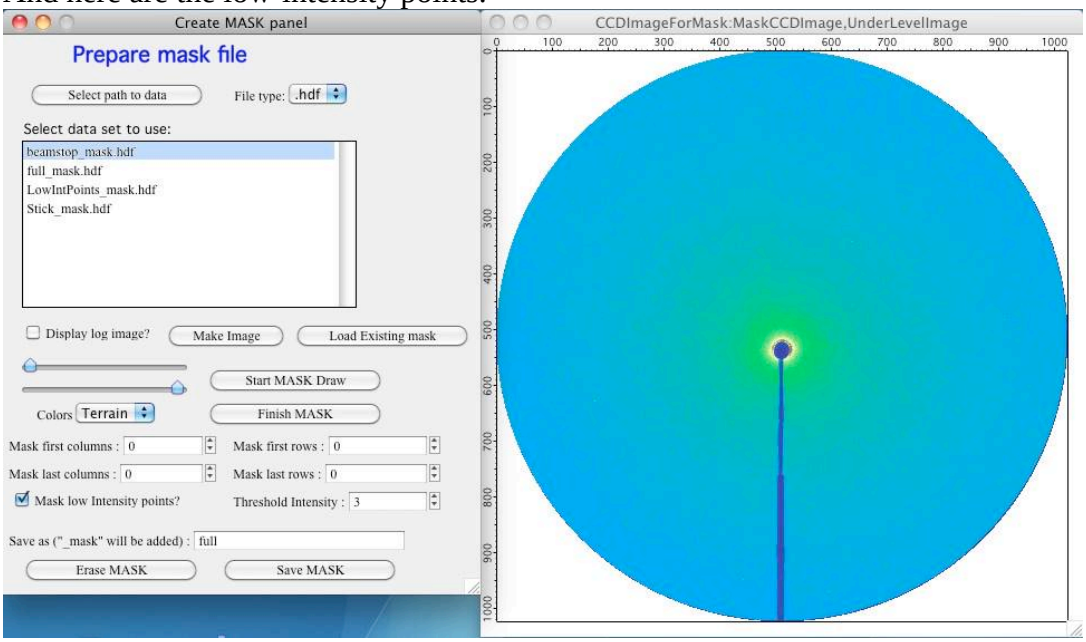
This is the beamstop:



This is the stick:

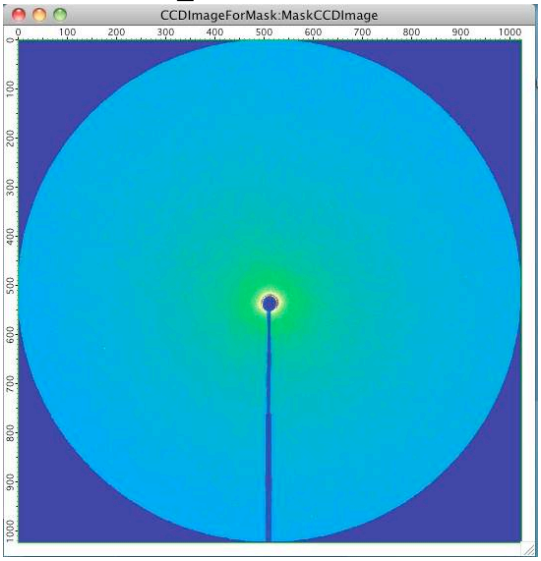


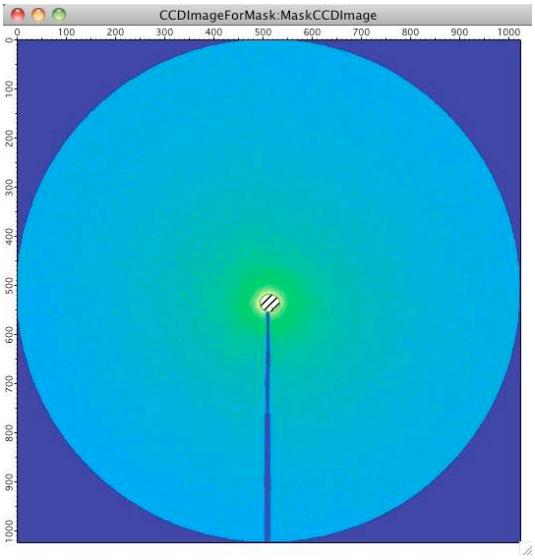
And here are the low-intensity points.

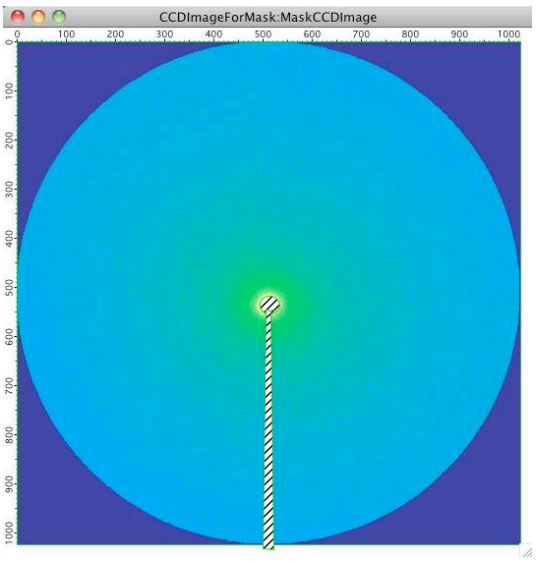


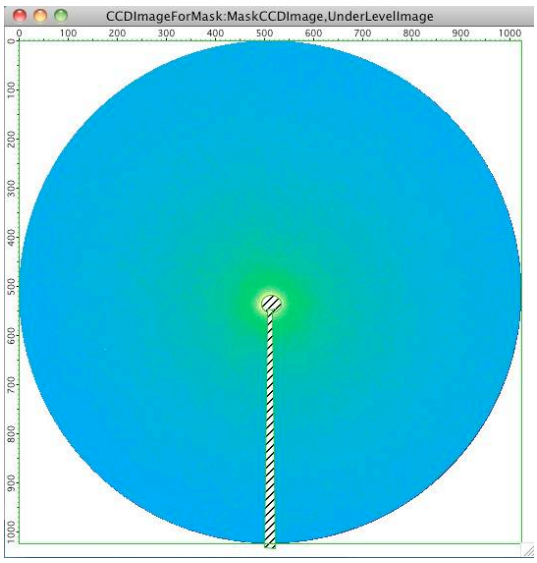
In each case I “Started MASK draw”, made my choices by drawing objects or modifying some settings, “Finished MASK”, and then gave the mask a name and saved it. The masks have extension .hdf and can be seen in the same place where the data are if you select File type hdf;

Now you can create new image using your data and load sequentially the three masks we just created - creating new more complex composite mask. You can also add more drawings to is as you see fit and then save complete "full" mask… Select one file after another in the list box and push button "Load existing mask".
Here are images with original, adding beamstop_mask.hdf, Stick_mask.hdf and then the LowIntPoints_mask.hdf:

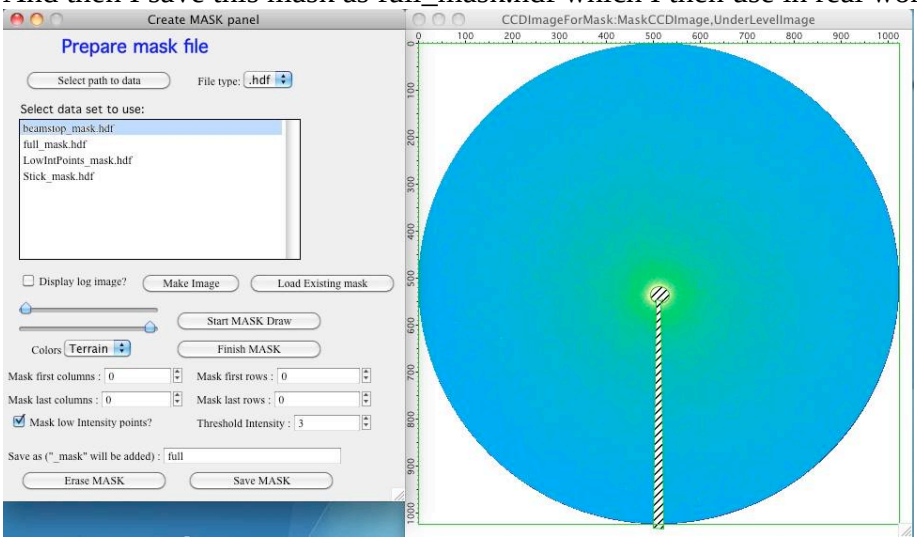






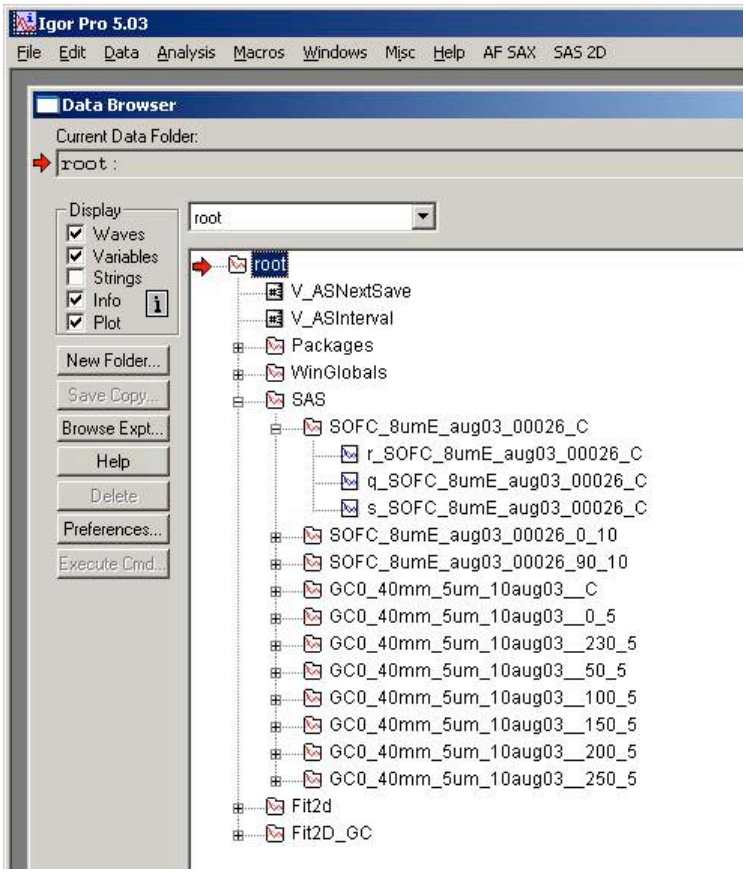


And then I save this mask as full_mask.hdf which I then use in real work…



Output data

Internal



Data are internally stored (if selected) within Igor experiment in folder root:SAS: in folders with
nameOfSample_C being the circular average
nameOfSample_Angle_halfWidth being sector average around direction Angle with sector half-width.

The wave names

X-axis data
q_NameOfSample_C (or _Angle_halfWidth) **q vector in A^{-1}**
t_ NameOfSample_C (or _Angle_halfWidth) **2 theta, if output with respect to 2 theta**
d_ NameOfSample_C (or _Angle_halfWidth) **d for output wrt d**
y axis data
r_ NameOfSample_C (or _Angle_halfWidth) **intensity (if calibrated in whatever units – thickness is converted to cm, so it should be cm^{-1})**
error
s_ NameOfSample_C (or _Angle_halfWidth) **error for intensity**

other
w_ NameOfSample_C (or _Angle_halfWidth) width of each bin of Q/d.2 theta. This is for LUT output, and provides data for bin-width smearing. Smaller number of bins, larger width of each. For linear binning, this is same number and is (Max-Min/numOfPoits), but for log binning this is varying function of bin position.

For Line profile data:
For example for GI_Vertical line in my test case, this was the name:
gc_saxs_395__GI_VLp_0.0077
"gc_saxs_395_"…. Part of the name of used image
GI_VLp_.... GI_Vertical Line
0.0077 …. q_y value at which the data were calculated.

Exported data are Int, error, Q, qx, qy, qz columns with header and column names
Saved data in Igor are

r_NameOfSample_ProfileIndicator_Qvalue	intensity
q_NameOfSample_ProfileIndicator_Qvalue	q [A^{-1}]
s_NameOfSample_ProfileIndicator_Qvalue	error
qy_NameOfSample_ProfileIndicator_Qvalue	qy [A^{-1}]
qz_NameOfSample_ProfileIndicator_Qvalue	qz [A^{-1}]
qx_NameOfSample_ProfileIndicator_Qvalue	qx [A^{-1}] (generated ONLY if GI… profile is used)

Note, intensity wave has attached wave note, containing some useful information:
CalibrationFormula=1*((Sa2D));CurrentMaskFileName=A mask_mask;QvectorNumberPoints=300;CircularAverage=1;

ASCII external

ASCII files with following data are stored in the selected folder:
CalibrationFormula=1*((Sa2D))
CurrentMaskFileName=A mask_mask
QvectorNumberPoints=300
AngularSector=150
AngularHalfWidth=5

0.01601654 0 0
0.0163735 1537 39.20459
0.01655496 1467 38.30144
0.01673842 1416 999.0073
0.01692392 1505 38.79433

The columns contain first q, second intensity and third error…

Instrument support

Instrument support are packages of specific additions in Nika to support special instrument. Depending on the instrument, these functions add capabilities and modify settings to make support of specific instrument easy.

DND CAT (APS 5ID) SAXS camera

DND CAT provides users with data, which are organized in specific folder structure. The data are reduced using scripts based on fit2d at the beamline. However, if users wants to process data later in different manner, they have to contact beamline staff and whole process is cumbersome.
Nika DND support is build on presence of evaluated data in text file, where header contains all necessary information for data reduction. Therefore, user opens this text file and the Tiff file with the processed image is found automatically (if user did not change the folder structure). Alternatively, user can point the Nika to the image files, when asked.
The data can then be reprocessed – for example different sectors can be analyzed etc. Note, that the user needs to make a new mask, but other parameters (beam center, wavelength, calibration constant s well as sample transmission and thickness) are loaded from the header.

To use:

1. select DND/txt as file type and point find data in the right folder. It is likely something like:…/APSCycle/YourName/Month/processing/data/plot_files
2. select and display one or more of the text files which contains current configuration and display. If the folder structure is correct, tiff image is found automatically. If not, Nika will ask for the image location. It should be necessary only once, unless the images are in different places.
3. Select “Instrument configurations” >> “DND CAT”. Select configuration from the listed names of the text fie(s) which were loaded already. This will set wavelength, pixel sizes, distance, and centers… Also it will set proper calibration configuration and functions, which should be used. Note: Dark was already subtracted from these files, so you need only the few parameters listed (thickness, calibration constant and transmission).
4. Create and use mask.
5. If you have empty run measurements, select the checkbox for “**subtract empty”**. Do not change values for “**Use I0/I0emp**” or value of 1 for I0. This is important to scale properly Empty and Sample incoming intensities and measurement times.
6. Select proper reduction parameters (circular, sector etc…).

15ID SAXS (pinSAXS/SAXS)

This is support for APS beamline 15ID SAXS and pinSAXS instruments. This is my beamline and there is special handout for how to use this tool.

SSRL Mat SAXS

This is support for SSRL Materials science SAXS camera. When selected, it sets fixed parameters for this instrument and also sets up lookup functions appropriate to read header values recorded in thsi image format.

TPA

This supports data from Australian SANS instrument. Not much more details provided yet and this code is not under development.

For other instrument scientists:
Other instrument setups can be added on request. Provide me with enough data and description and I can write support for your instrument.

Create Flood field

This simple routine will help user to create “Flood field” – or as NIKA is calling Pix2D sensitivity field. This routine was revamped and upgraded in version 1.51 with more functionality.

Definition: This field is matrix of real values such, that when intensity measured in any given pixel is DIVIDED of its pix2D value, the intensity is scaled to correct intensity.

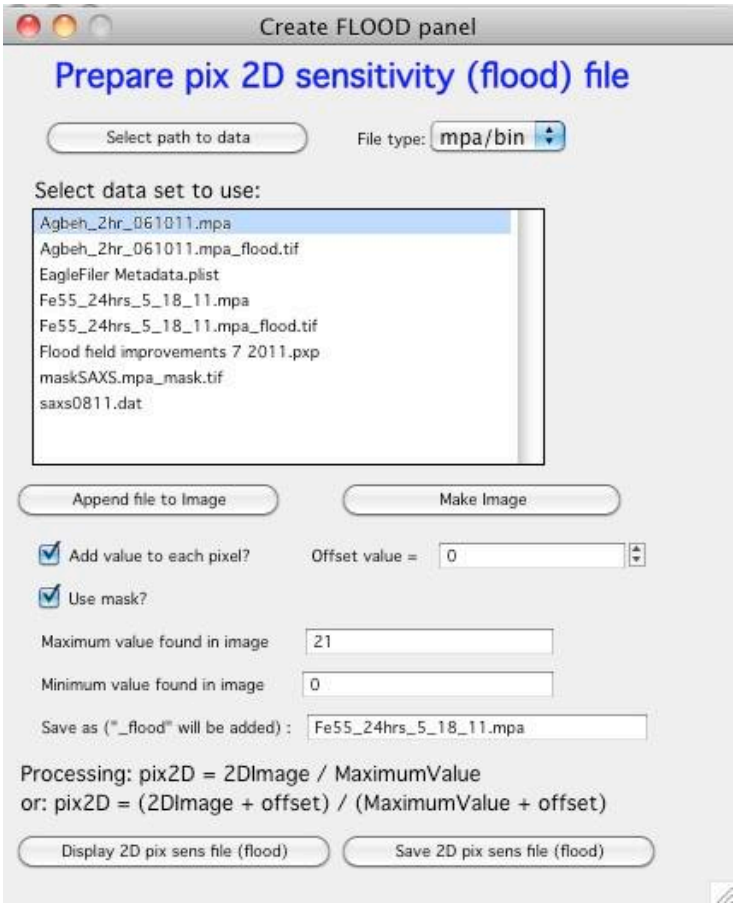
Corrected value = Measured value / pix2D sensitivity value

To create this pixel sensitivity map one has to expose all pixels to the same incoming flux of photons (neutrons) – aka flood field, flat field etc… With enough counts measured for each pixel – and assuming the incoming field is really “flat” – the intensity measured for each pixel is now inversely proportional to pixel sensitivity.

If you have such image, this routine can help to convert this image into NIKA-type pix2D sensitivity image. Note, that this image is saved as tiff file with single precision real number in each pixel – this image may not be readable by other packages!!!!

To create this pix2D sensitivity map, one needs to load in image and for each pixel divide measured value for that pixel by the maximum intensity measured in the image. If any pixel has intensity = 0 (WRONG… You need to collect at least 1 count in each point) it is possible to add 1 to each point intensity to avoid having 0 as corrections in the pix2D sensitivity file, which would cause dividing by 0 and therefore would remove this pixel from any calculations in the future….

GUI



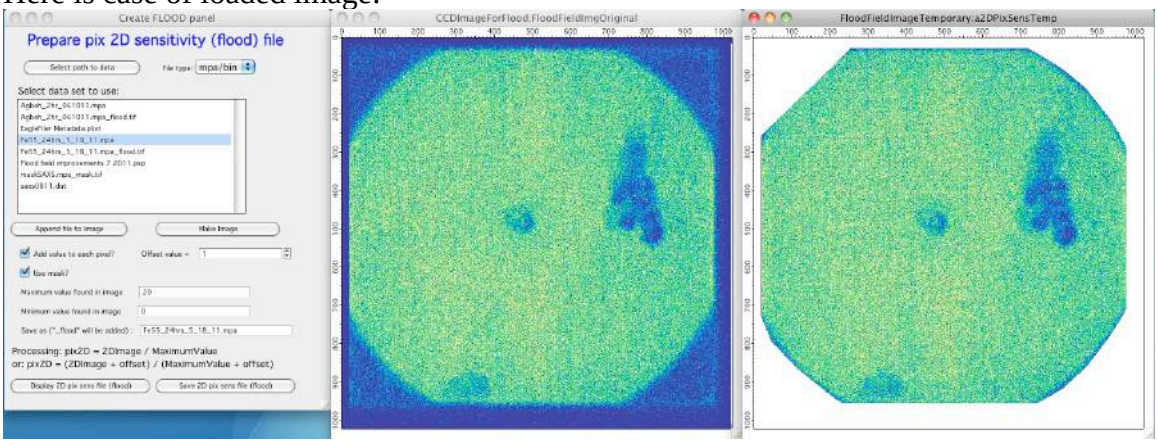
The GUI is very similar to Mask GUI:
"*select path to data"* Select path, where measured flood field is located. Note, resulting pix2D sensitivity file will also be there.
"*File type*" well, select type of data you have. Note, resulting pix2D sensitivity file is tiff file.

Note, at this time this tool can read ONLY one file at time. If you have multiple files to process, you can append the images with the button "*Append file to image*" and the intensities will be added to existing image. If the 2D image does not exist, new image is created. However, if you need to start fresh you need to use button "*Make Image"* which will create new 2DF image to work on.

"*Select data set to use"* select the image to use
"*Make Image*" Loads data in and Creates image
"*Add value to each point"* – select, if you need to add some number of counts to each point in image. This will open new control "*Offset to add to each point"*.
Note, the offset will be selected automatically, if during loading the data this tool detects, that there are points with 0 intensity in them. Conversely, this option will be deselected, if minimum intensity in the image is more than 0….

"*Use Mask"*- mask loaded in the main tool is used for Pix sensitivity development. See below.
Further controls will be populated during loading of the data set:
"*Maximum value found in image*" and "*Minimum value found in image*". These are here mainly for information.
However, note, that it is possible to overwrite the "*Maximum value found in image"* and this new value will be used instead of the original maximum value for correction described above. It is therefore possible to create pix2D sensitivity file with different range of values. Note, that the code does not check for range of pix2D sensitivity values…
Warning
The code really does not check for values here, and it is possible to set offset negative, for example, and generate negative pix2D sensitivity map… This will cause all sorts of problems later on – **you were warned!**

Here is case of loaded image:



Left is panel, middle is image loaded in for creating the pix sensitivity map and right is calculated pix sensitivity map – after masking and scaling by max intensity found (or user input in the panel). Note, that as you change the values in the panel the imegase should change to reflect them. In case they do not, "*Display …"* button will update these images and recreate, if necessary.
"*Save 2D pix sensitivity file (flood)"* will process the file as described above and save it. It will be saved in place where original file came from and _flood.tif will be added to text in the "*Save as ("_flood" will be added)"* control. The text will be first checked to be acceptable name for OS.