WAND^2 Single Crystal Diffraction Scripts Manual

September 2025
Xiao Hu
Neutron Scattering Scientist
Oak Ridge National Laboratory

Index

I. Prologue

This manual provides an overview of the general workflow of **WAND^2 single-crystal data analysis** using Mantid-based scripts. The workflow consists of the following stages:

- **Data loading**
- **Data slicing**
- **Peak analysis**
    - Peak search, indexing, filtering
    - Peak integration

The **peak analysis** section describes how ready-for-refinement peak files – containing indices, intensities, and associated errors – are generated with Mantid-based scripts. Additional details on the underlying script mechanisms are included in the Appendix.

Two methods of peak integration are included:

- **Method 1 (Conventional Mantid algorithm)**
  Utilize Mantid's built-in functions (e.g., IntegratePeaksMD). This method is well suited for rapid integration and quick peak inspection. However, its error propagation can be problematic.

- **Method 2 (Detector image-based algorithm)**
  A newly developed algorithm that performs peak integration directly from detector images. Although more sophisticated than Method 1, this approach resolves several limitations, such as handling adjacent peaks, irregular peak profiles, and aluminum powder ring contamination.

For general graphical user interface (GUI) instructions, please refer to the Remote Desktop Analysis platform at:

data → HFIR → HB2C → shared → WANDscripts → manual

and read the file *SCD reduction manual.pdf*.

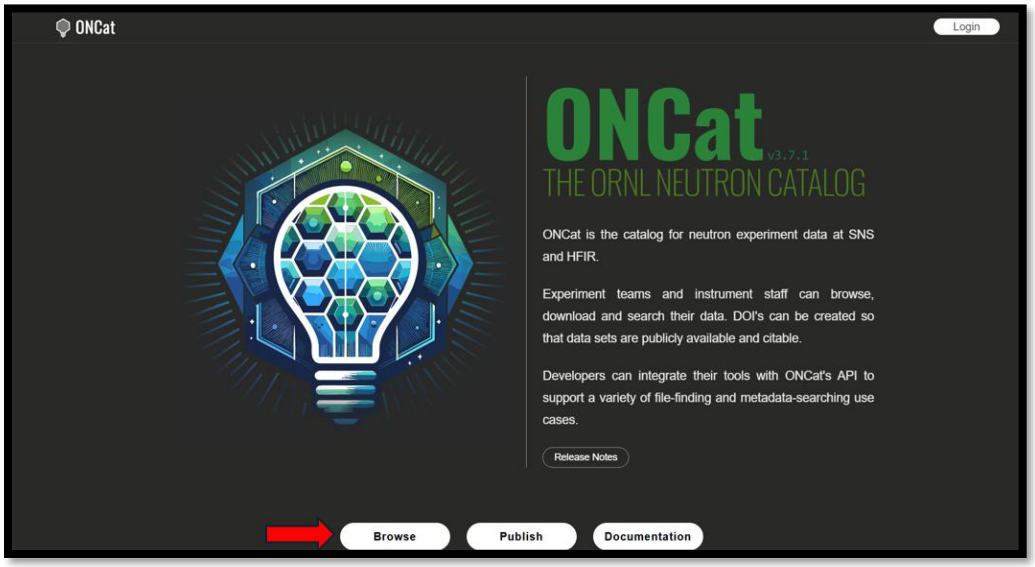
If you have any questions or feedback about this manual, please contact hux6@ornl.gov.

Back to Index

II. Locate IPTS and Run numbers & Open Mantid Workbench

(a) Find experimental data under proposal number (IPTS)

(1) Go to https://oncat.ornl.gov/ to view your experiment information. Click "Browse" and log in to ONCat catalog with UCAMS.

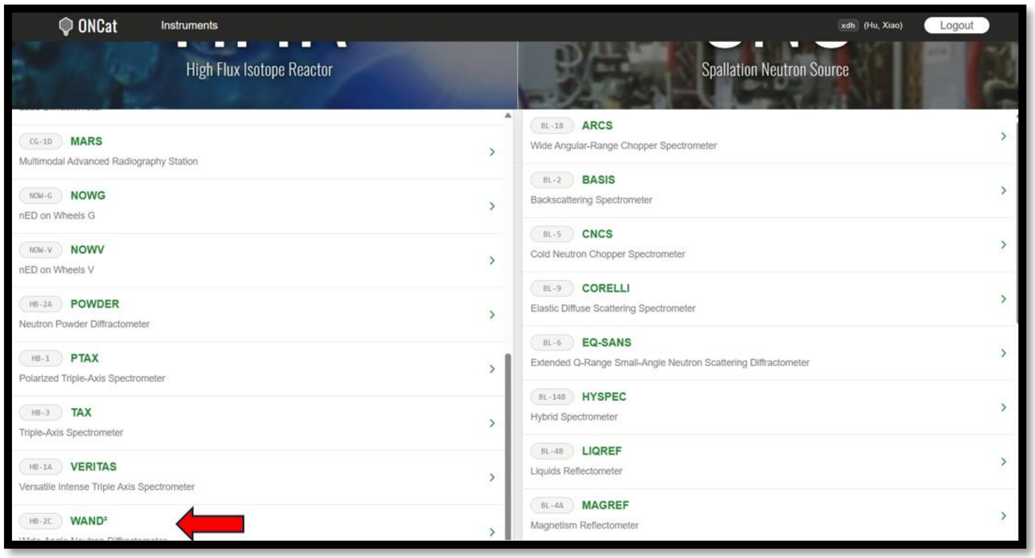


ONCat is the catalog for neutron experiment data at SNS and HFIR.

Experiment teams and instrument staff can browse, download and search their data. DOI's can be created so that data sets are publicly available and citable.

Developers can integrate their tools with ONCat's API to support a variety of file-finding and metadata-searching use cases.

(2) Find the instrument in the HFIR/SNS instrument list and click on WAND² .



CG-1D MARS
Multimodal Advanced Radiography Station

NOW-G NOWG
nED on Wheels G

NOW-V NOWV
nED on Wheels V

HB-2A POWDER
Neutron Powder Diffractometer

HB-1 PTAX
Polarized Triple-Axis Spectrometer

HB-3 TAX
Triple-Axis Spectrometer

HB-1A VERITAS
Versatile Intense Triple Axis Spectrometer

HB-2C WAND²

BL-18 ARCS
Wide Angular-Range Chopper Spectrometer

BL-2 BASIS
Backscattering Spectrometer

BL-5 CNCS
Cold Neutron Chopper Spectrometer

BL-9 CORELLI
Elastic Diffuse Scattering Spectrometer

BL-6 EQ-SANS
Extended Q-Range Small-Angle Neutron Scattering Diffractometer

BL-14B HYSPEC
Hybrid Spectrometer

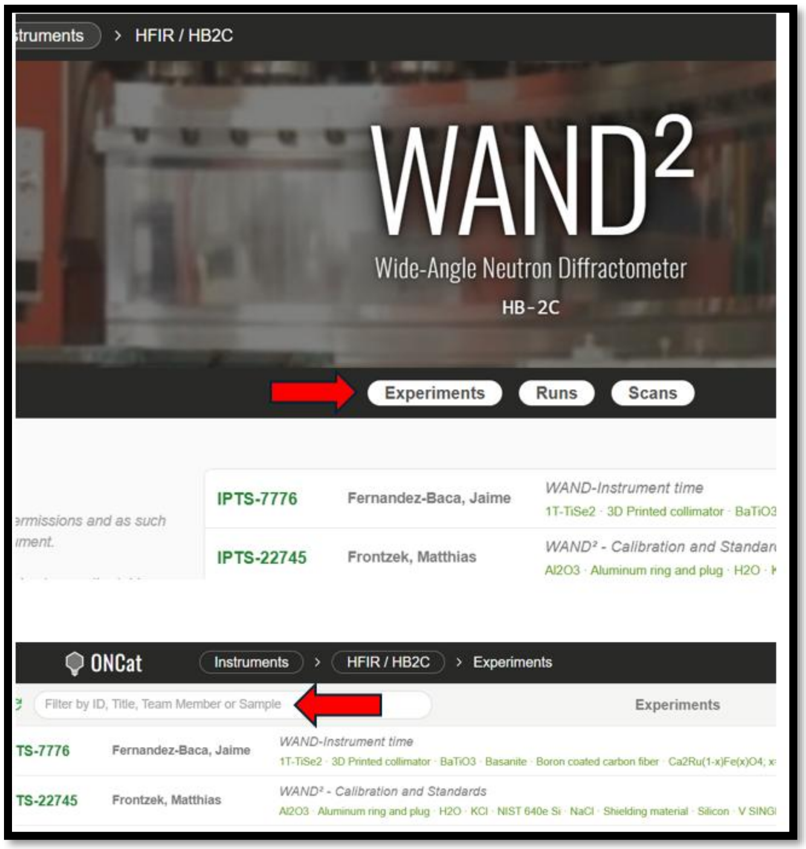
BL-4B LIQREF
Liquids Reflectometer

BL-4A MAGREF
Magnetism Reflectometer

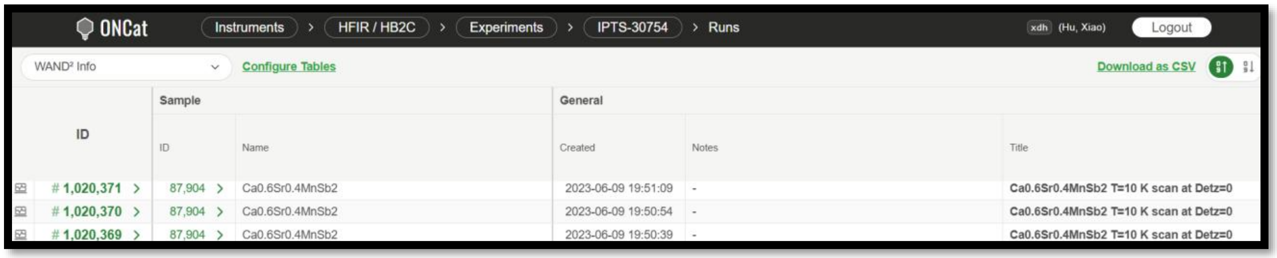
Back to Index

(3) Search for experiment data through: Experiments → type in proposal number (IPTS).

Scans will be displayed along with run number (ID), sample information, and scan descriptions. Record run numbers you need for analysis.



IPTS-7776	Fernandez-Baca, Jaime	WAND-Instrument time 1T-TiSe2 · 3D Printed collimator · BaTiO3 · Basanite · Boron coated carbon fiber · Ca2Ru(1-x)Fe(x)O4; x
IPTS-22745	Frontzek, Matthias	WAND² - Calibration and Standards Al2O3 · Aluminum ring and plug · H2O · KCl · NIST 640e Si · NaCl · Shielding material · Silicon · V SINGl



	Sample		General		
ID	ID	Name	Created	Notes	Title
# 1,020,371	87,904	Ca0.6Sr0.4MnSb2	2023-06-09 19:51:09	-	Ca0.6Sr0.4MnSb2 T=10 K scan at Detz=0
# 1,020,370	87,904	Ca0.6Sr0.4MnSb2	2023-06-09 19:50:54	-	Ca0.6Sr0.4MnSb2 T=10 K scan at Detz=0
# 1,020,369	87,904	Ca0.6Sr0.4MnSb2	2023-06-09 19:50:39	-	Ca0.6Sr0.4MnSb2 T=10 K scan at Detz=0

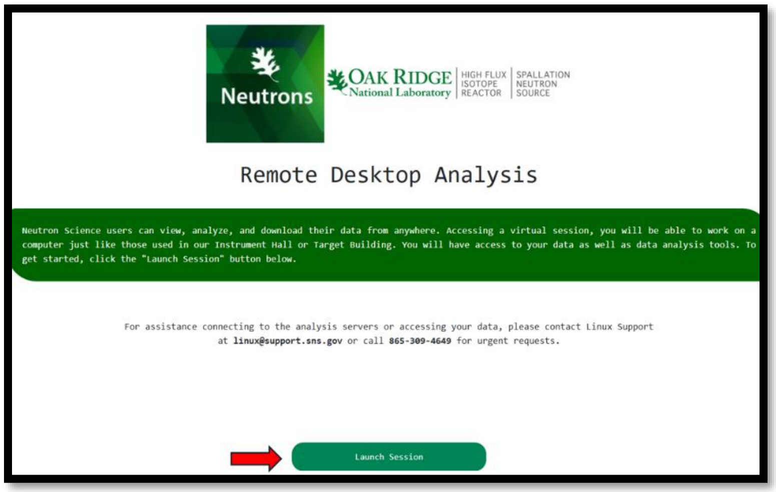
Note:

Vanadium calibration was usually performed before or at the beginning of each cycle. Users have access to the vanadium IPTS 23858. Use the vanadium run number that is closet (in timeline) to your beamtime.

Back to Index

(b) Open Mantid Workbench in analysis cluster

(1) Go to Remote Desktop Analysis at https://analysis.sns.gov/ and log in with UCAMS username and password.

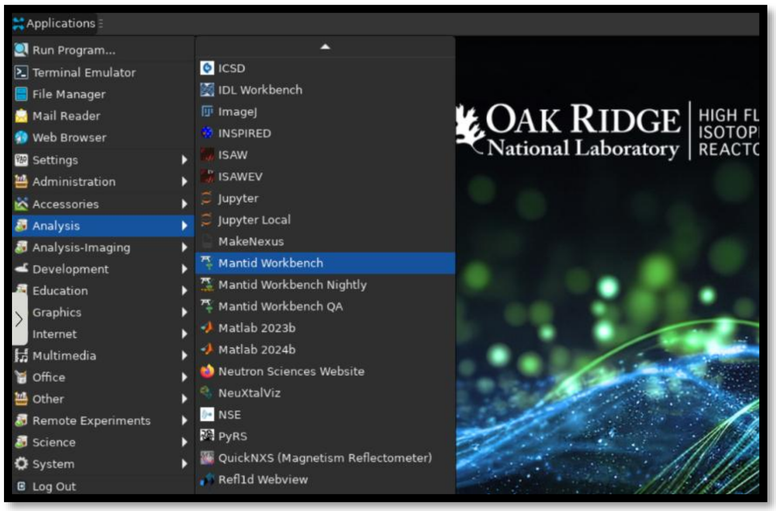

Neutrons
OAK RIDGE National Laboratory
HIGH FLUX ISOTOPE REACTOR
SPALLATION NEUTRON SOURCE
Launch Session

Remote Desktop Analysis

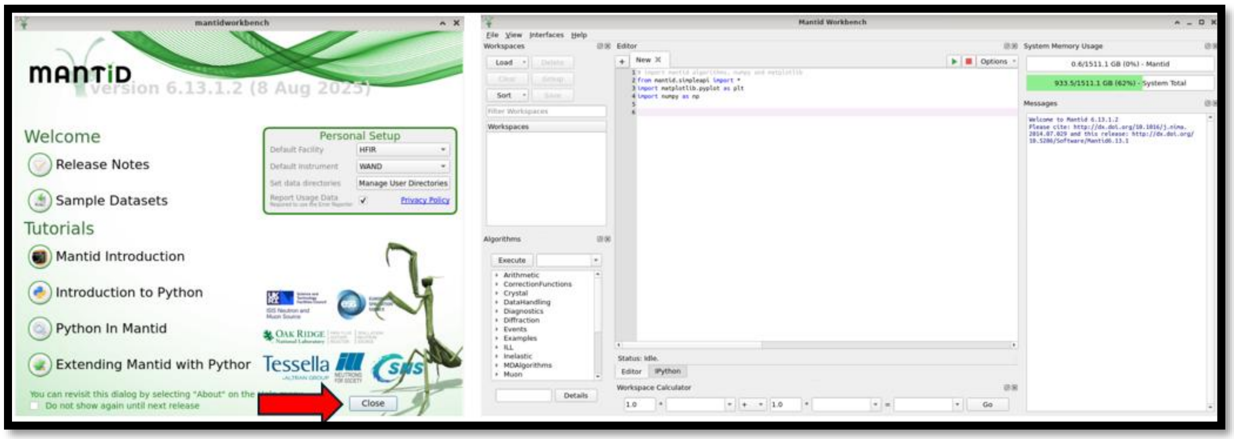
Neutron Science users can view, analyze, and download their data from anywhere. Accessing a virtual session, you will be able to work on a computer just like those used in our Instrument Hall or Target Building. You will have access to your data as well as data analysis tools. To get started, click the "Launch Session" button below.

For assistance connecting to the analysis servers or accessing your data, please contact Linux Support at **linux@support.sns.gov** or call **865-309-4649** for urgent requests.

(2) Once logged in, on the left upper corner, go to Application → Analysis → Mantid Workbench.



(3) When Mantid Workbench opens, there will be an option to personalize the setup and tutorial available if needed. If a personalized setup is not necessary, close the Welcome window. You should see the Mantid graphical user interface on your screen.



```
# import mantid algorithms, numpy and matplotlib
from mantid.simpleapi import *
import matplotlib.pyplot as plt
import numpy as np
```

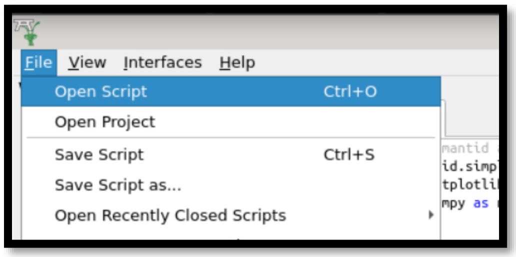
```
Welcome to Mantid 6.13.1.2
Please cite: http://dx.doi.org/10.1016/j.nima.
2014.07.029 and this release: http://dx.doi.org/
10.5286/Software/Mantid6.13.1
```

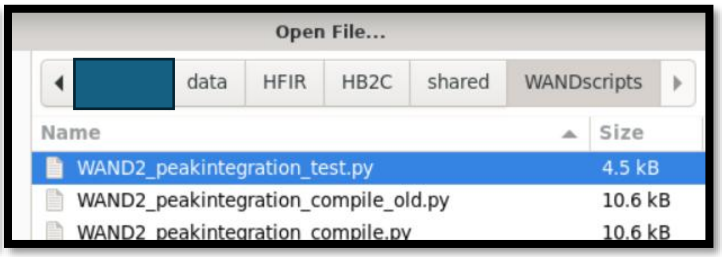
Back to Index

III. Script execution in Mantid

(a) Open script in Mantid

Go to **File** on the left upper corner of the Mantid interface and hit "**Open Script**" or **Ctrl + O**. Locate and open the target script.

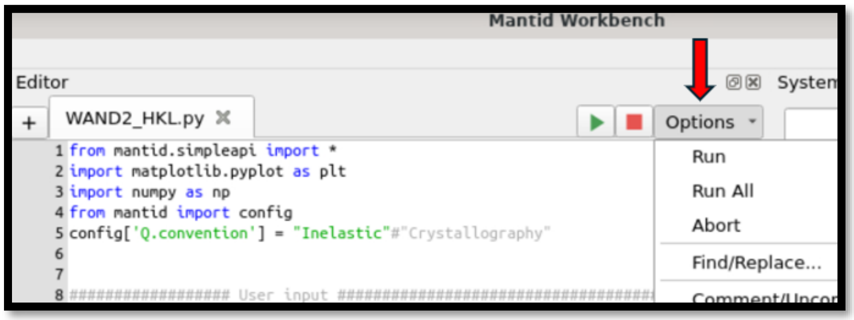




Name	Size
WAND2_peakintegration_test.py	4.5 kB
WAND2_peakintegration_compile_old.py	10.6 kB
WAND2_peakintegration_compile.py	10.6 kB

(b) Execute (entire or part of) the script

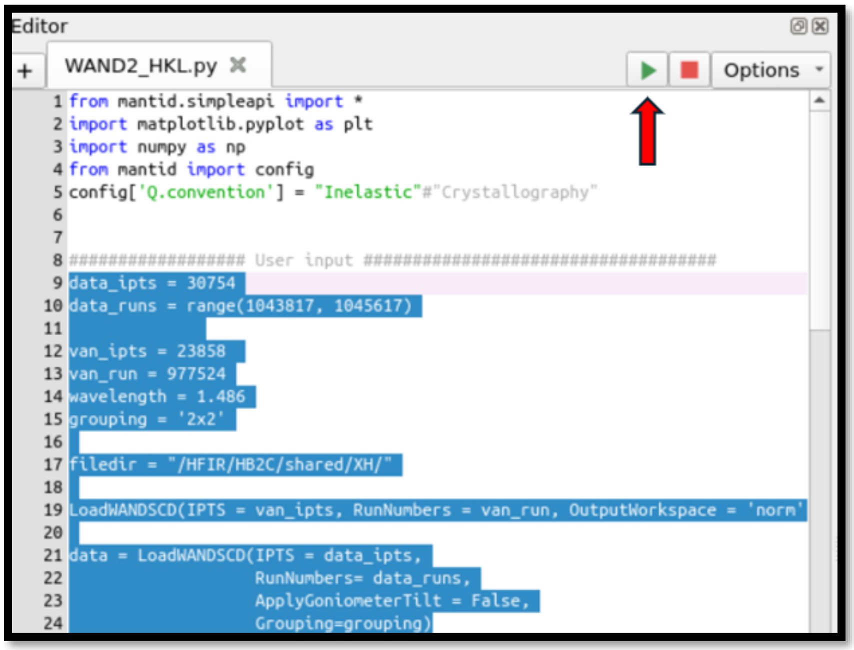
(1) To execute the entire script, click **Options** on the upper right corner and select **Run All.**



```
from mantid.simpleapi import *
import matplotlib.pyplot as plt
import numpy as np
from mantid import config
config['Q.convention'] = "Inelastic"#"Crystallography"


################## User input ####################################
```

(2) To execute part of the script, highlight the target part and click the **Run** button (**green triangle**).



```
from mantid.simpleapi import *
import matplotlib.pyplot as plt
import numpy as np
from mantid import config
config['Q.convention'] = "Inelastic"#"Crystallography"


################## User input ####################################
data_ipts = 30754
data_runs = range(1043817, 1045617)

van_ipts = 23858
van_run = 977524
wavelength = 1.486
grouping = '2x2'

filedir = "/HFIR/HB2C/shared/XH/"

LoadWANDSCD(IPTS = van_ipts, RunNumbers = van_run, OutputWorkspace = 'norm'

data = LoadWANDSCD(IPTS = data_ipts,
                   RunNumbers= data_runs,
                   ApplyGoniometerTilt = False,
                   Grouping=grouping)
```

Back to Index

IV. Data slicing

There are two options for data slicing, involving two algorithms, respectively:

- **ConvertQtoHKLMDHisto**
- **ConvertWANDSCDtoQ**.

ConvertQtoHKLMDHisto uses data workspace (in reciprocal Q-space) generated during peak analysis, but it does not deal with vanadium normalization properly. Thus, the other method involving **ConvertWANDSCDtoQ** is introduced here:

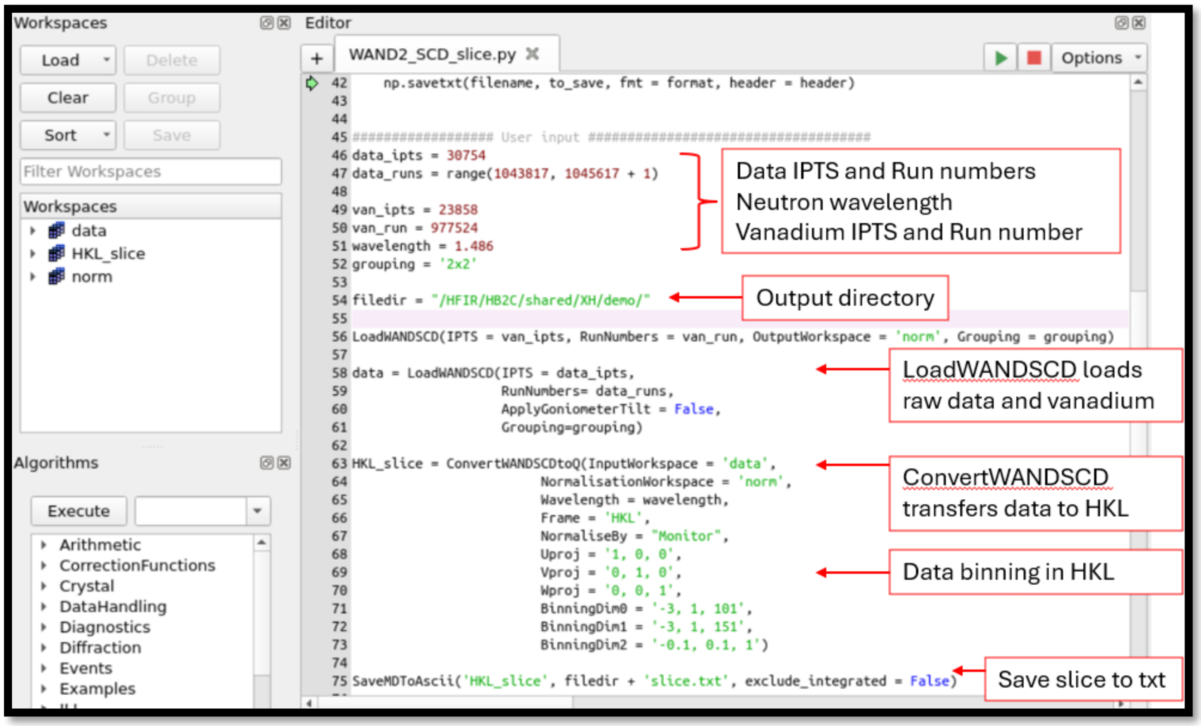
(1) Follow the path "data → HFIR → HB2C → shared → WANDscripts" and locate the script

WAND2_SCD_slice.py

(2) Copy and paste it to your working directory.
(3) Open the script in Mantid by going to **File** on the left upper corner of the Mantid interface and hitting "**Open Script**" or **Ctrl + O**. Locate your working directory and select the script.

(4) The script looks like below. Command lines before line 43 define a file-saving function **SaveMDToAscii**. It saves the output to the customized directory in ASCII format.

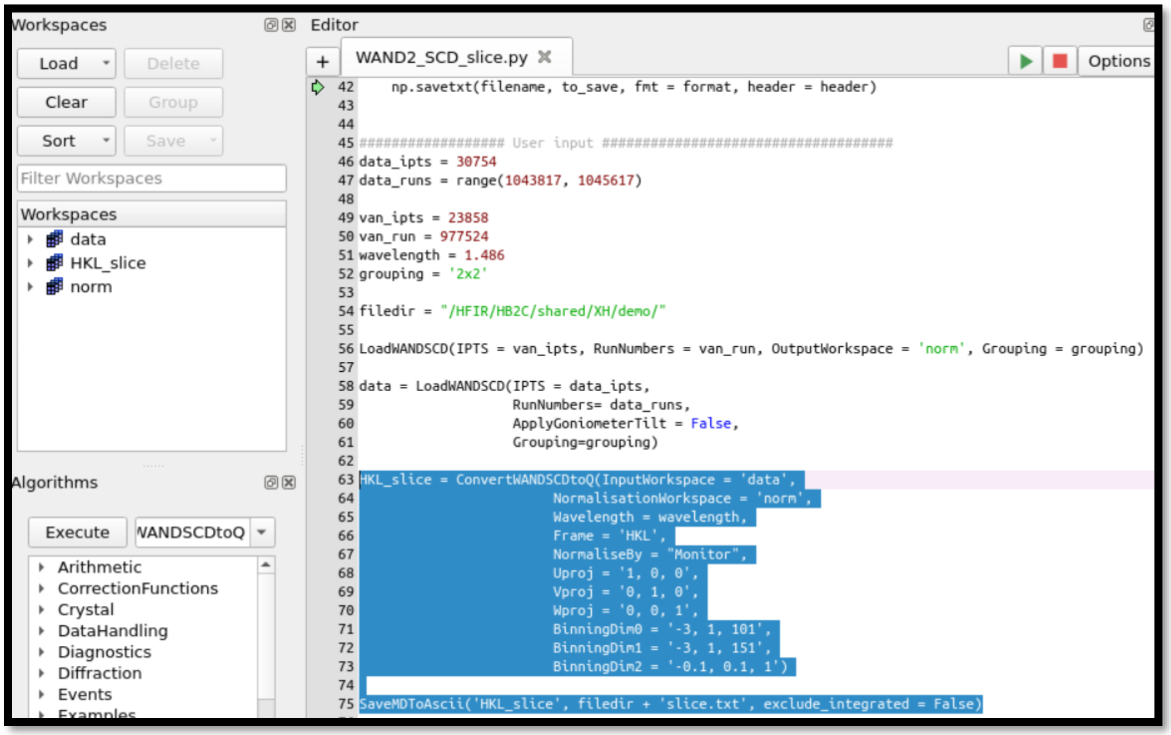
(5) Modify relevant parameters (explained in the *Note* below) and execute the entire script by going to **Options** on the upper right corner and selecting **Run All**.



```
42     np.savetxt(filename, to_save, fmt = format, header = header)
43
44
45 ################## User input ####################################
46 data_ipts = 30754
47 data_runs = range(1043817, 1045617 + 1)
48
49 van_ipts = 23858
50 van_run = 977524
51 wavelength = 1.486
52 grouping = '2x2'
53
54 filedir = "/HFIR/HB2C/shared/XH/demo/"
55
56 LoadWANDSCD(IPTS = van_ipts, RunNumbers = van_run, OutputWorkspace = 'norm', Grouping = grouping)
57
58 data = LoadWANDSCD(IPTS = data_ipts,
59                    RunNumbers= data_runs,
60                    ApplyGoniometerTilt = False,
61                    Grouping=grouping)
62
63 HKL_slice = ConvertWANDSCDtoQ(InputWorkspace = 'data',
64                               NormalisationWorkspace = 'norm',
65                               Wavelength = wavelength,
66                               Frame = 'HKL',
67                               NormaliseBy = "Monitor",
68                               Uproj = '1, 0, 0',
69                               Vproj = '0, 1, 0',
70                               Wproj = '0, 0, 1',
71                               BinningDim0 = '-3, 1, 101',
72                               BinningDim1 = '-3, 1, 151',
73                               BinningDim2 = '-0.1, 0.1, 1')
74
75 SaveMDToAscii('HKL_slice', filedir + 'slice.txt', exclude_integrated = False)
```

Back to Index

(6) To revise the sliced data (e.g., binning size, Frame), modify relevant parameters in the **ConvertWANDSCDtoQ** command lines, highlight the relevant command lines (as shown below for example), and hit **Run** button (**green triangle** button) on the upper right corner.



```
    np.savetxt(filename, to_save, fmt = format, header = header)


################## User input ####################################
data_ipts = 30754
data_runs = range(1043817, 1045617)

van_ipts = 23858
van_run = 977524
wavelength = 1.486
grouping = '2x2'

filedir = "/HFIR/HB2C/shared/XH/demo/"

LoadWANDSCD(IPTS = van_ipts, RunNumbers = van_run, OutputWorkspace = 'norm', Grouping = grouping)

data = LoadWANDSCD(IPTS = data_ipts,
                   RunNumbers= data_runs,
                   ApplyGoniometerTilt = False,
                   Grouping=grouping)

HKL_slice = ConvertWANDSCDtoQ(InputWorkspace = 'data',
                        NormalisationWorkspace = 'norm',
                        Wavelength = wavelength,
                        Frame = 'HKL',
                        NormaliseBy = "Monitor",
                        Uproj = '1, 0, 0',
                        Vproj = '0, 1, 0',
                        Wproj = '0, 0, 1',
                        BinningDim0 = '-3, 1, 101',
                        BinningDim1 = '-3, 1, 151',
                        BinningDim2 = '-0.1, 0.1, 1')

SaveMDToAscii('HKL_slice', filedir + 'slice.txt', exclude_integrated = False)
```

Note:

- *Users need to input the **data IPTS** and **run numbers**, along with the **vanadium IPTS** and **run number**. Data run numbers are in the format of a python list. It could be*
  - ***range(start, end + 1)** (as shown in the screenshot above)*
  - ***[run1, run2, run3, …]***
  - ***list(range(start1, end1 + 1)) + list(range(start2, end2 + 1))***

- ***wavelength** is the incident neutron beam wavelength. **grouping** is for faster data processing. It has options: "2x2", "4x4", "None". **filedir** is the directory path to save output files. Modify it to your working directory.*

- ***LoadWANDSCD** loads data and vanadium files into Mantid workspaces, **data** and **norm** here, which will appear in the left-hand side "Workspace" window after execution. Modify the name of the **OutputWorkspace** if necessary.*

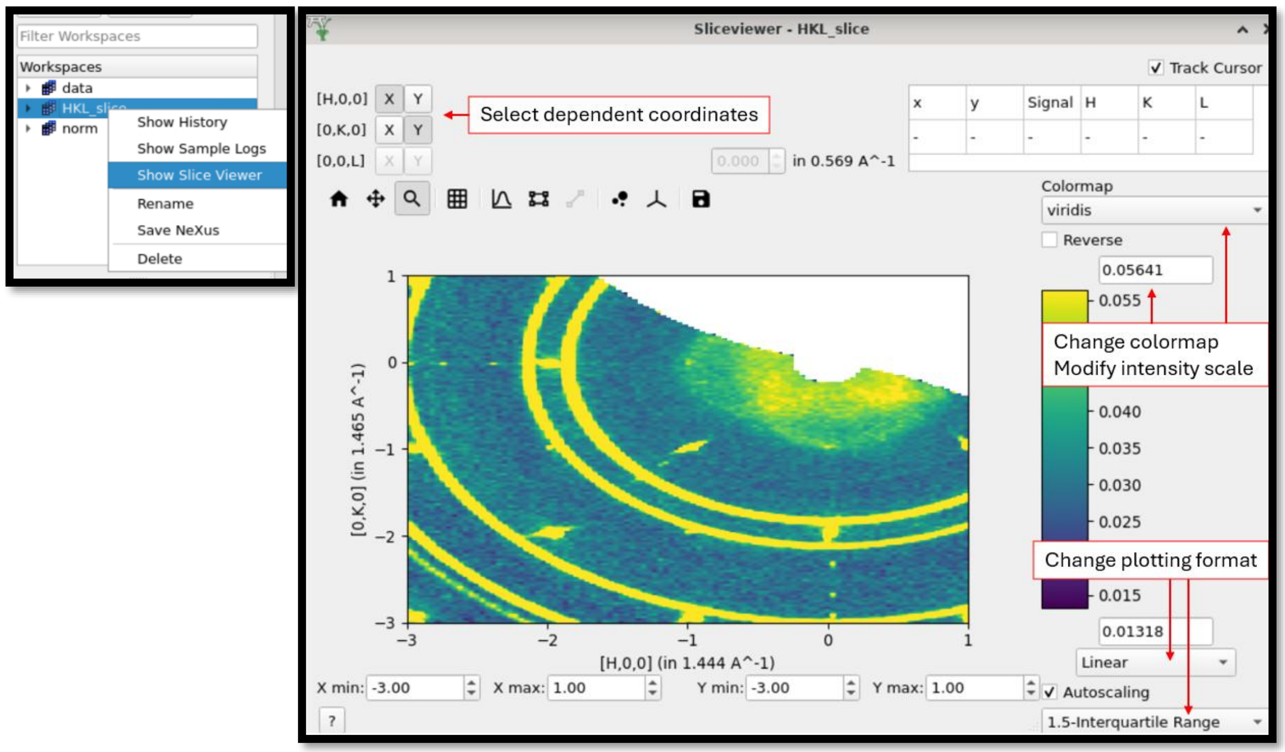
- ***ConvertWANDSCDtoQ** will bin and slice data in a customized style.*
  - *Input data and vanadium workspace. (vanadium is for normalization)*

Back to Index

- ***Frame*** *can be "HKL" or "Q_sample". "Q_sample" has a unit of* $\mathrm{\AA}^{-1}$.
- ***NormalizeBy*** *has multiple choices: "Monitor", "Time", "None".*
- ***Uproj, Vproj, Wproj*** *are basis vectors defining data binning dimensions.*
- ***BinningDim0,1,2*** *refer to data binning sizes along* ***Uproj, Vproj, Wproj,*** *respectively. Format:* ***"start, end, number of points"****.*

- ***SaveMDtoAscii*** *saves output sliced data into ASCII files. In the example, the sliced data is the workspace* ***HKL_slice.***

(3) To view the sliced data in Mantid, right-click the data workspace in the "Workspaces" window on the left-hand side and select **Show Slice Viewer**. A slice viewer window will pop up (shown below).



Note:
The ***Show Slice Viewer*** *visualizes the data in the reciprocal space, enables fast data inspection, and helps users do further data slicing/cutting. Move the mouse to each button and icon to see the pop-up illustration of its functionality.*

Back to Index

V. Peak analysis

(a) Data loading

(1) Follow the path "data → HFIR → HB2C → shared → WANDscripts" and locate the script

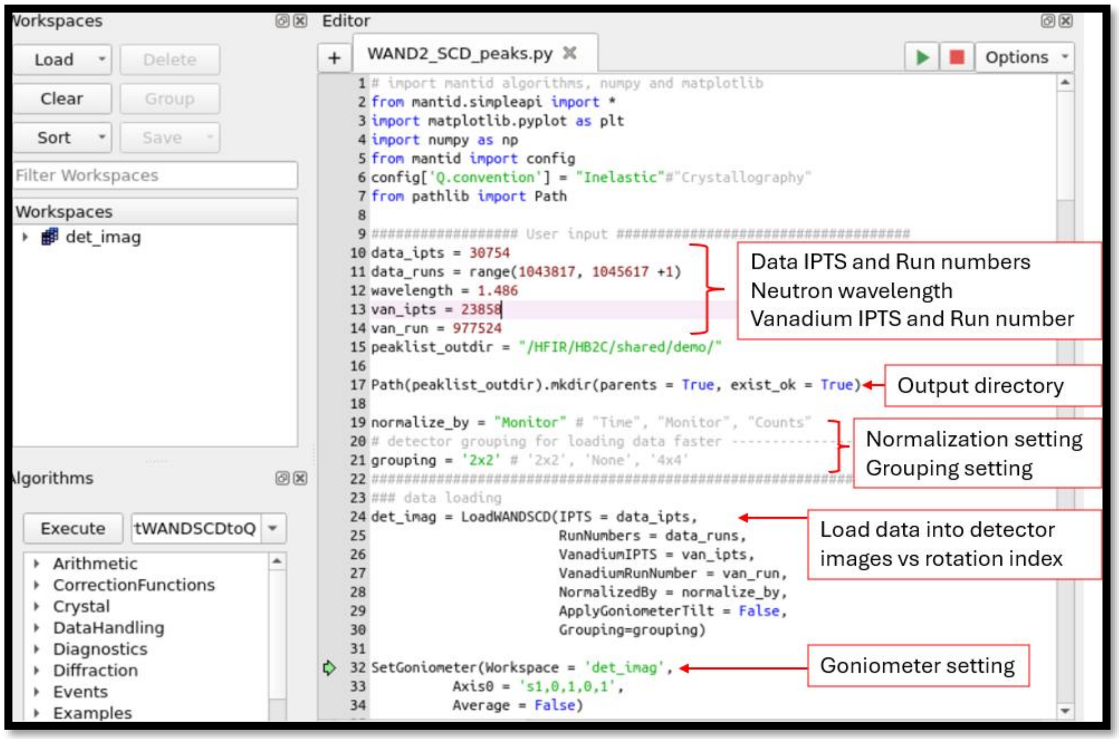
WAND2_SCD_peaks.py

(2) Copy and paste it to your working directory.

(3) Open the script in Mantid by going to **File** on the left upper corner of the Mantid interface and hitting "**Open Script**" or **Ctrl + O**. Locate your working directory and select the script. The script looks like below.

(4) Input data IPTS and run numbers, and incident neutron wavelength.
(5) Input vanadium IPTS 23858 and run number.
(6) Input the output directory path (to save output files).
(7) Choose **normalize_by**, "Monitor" or "Time" or "None". "Monitor" and "Time" are commonly used.

(8) Choose **grouping**, "2x2" or "4x4" or "None". It is recommended to use "2x2".
(9) Highlight command lines (line 1 to 34) and hit **Run** (**green triangle**).



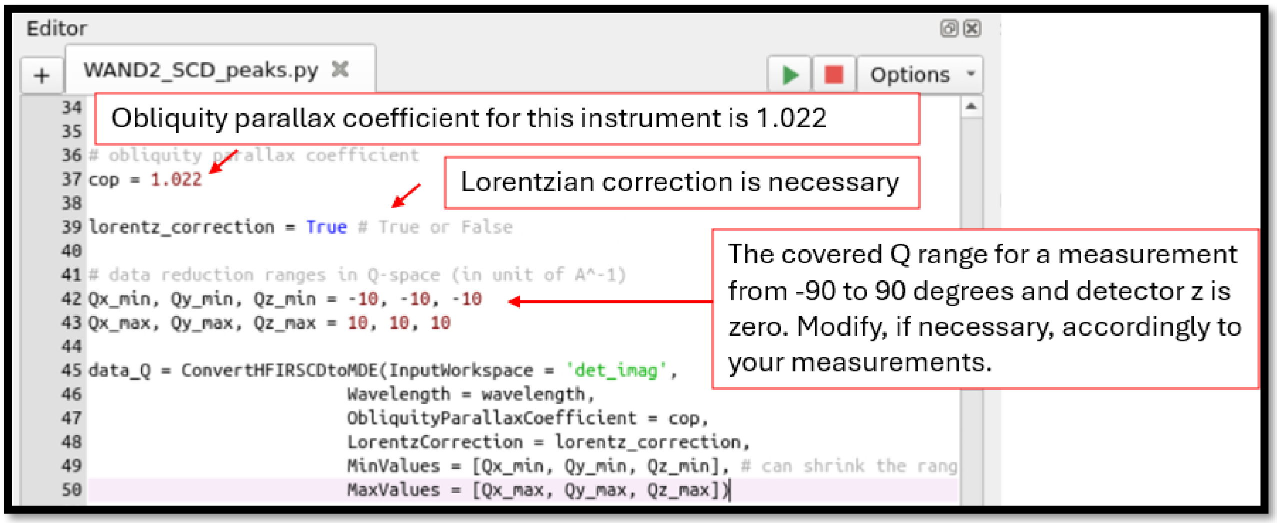
```
 1 # import mantid algorithms, numpy and matplotlib
 2 from mantid.simpleapi import *
 3 import matplotlib.pyplot as plt
 4 import numpy as np
 5 from mantid import config
 6 config['Q.convention'] = "Inelastic"#"Crystallography"
 7 from pathlib import Path
 8
 9 ################## User input ####################################
10 data_ipts = 30754
11 data_runs = range(1043817, 1045617 +1)
12 wavelength = 1.486
13 van_ipts = 23858
14 van_run = 977524
15 peaklist_outdir = "/HFIR/HB2C/shared/demo/"
16
17 Path(peaklist_outdir).mkdir(parents = True, exist_ok = True)
18
19 normalize_by = "Monitor" # "Time", "Monitor", "Counts"
20 # detector grouping for loading data faster -------------
21 grouping = '2x2' # '2x2', 'None', '4x4'
22 ####################################################################
23 ### data loading
24 det_imag = LoadWANDSCD(IPTS = data_ipts,
25                        RunNumbers = data_runs,
26                        VanadiumIPTS = van_ipts,
27                        VanadiumRunNumber = van_run,
28                        NormalizedBy = normalize_by,
29                        ApplyGoniometerTilt = False,
30                        Grouping=grouping)
31
32 SetGoniometer(Workspace = 'det_imag',
33          Axis0 = 's1,0,1,0,1',
34          Average = False)
```

Back to Index

Note:

- ***LoadWANDSCD** loads data files into Mantid workspace, **det_imag**, as shown in the "Workspaces" window on the left-hand side of the interface. Modify workspace name as needed.*
- ***SetGoniometer** defines the goniometer rotation axis → s1 motor rotation, about the vertical axis. Horizontal plane is the scattering plane at WAND².*

(b) Convert data into reciprocal Q-space



Obliquity parallax coefficient for this instrument is 1.022

Lorentzian correction is necessary

The covered Q range for a measurement from -90 to 90 degrees and detector z is zero. Modify, if necessary, accordingly to your measurements.

```
34
35
36 # obliquity parallax coefficient
37 cop = 1.022
38
39 lorentz_correction = True # True or False
40
41 # data reduction ranges in Q-space (in unit of A^-1)
42 Qx_min, Qy_min, Qz_min = -10, -10, -10
43 Qx_max, Qy_max, Qz_max = 10, 10, 10
44
45 data_Q = ConvertHFIRSCDtoMDE(InputWorkspace = 'det_imag',
46                              Wavelength = wavelength,
47                              ObliquityParallaxCoefficient = cop,
48                              LorentzCorrection = lorentz_correction,
49                              MinValues = [Qx_min, Qy_min, Qz_min], # can shrink the rang
50                              MaxValues = [Qx_max, Qy_max, Qz_max])
```

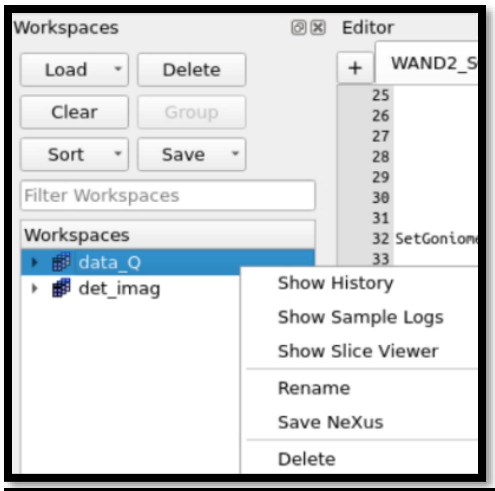
(1) Set the obliquity parallax coefficient.
(2) Set **lorentz_correction** = True.
(3) Set Q-range for peak search. Unit: Å^{-1}.
(4) Highlight command lines (line 37 to 50) shown below and hit **Run** (**green triangle**).

Note:

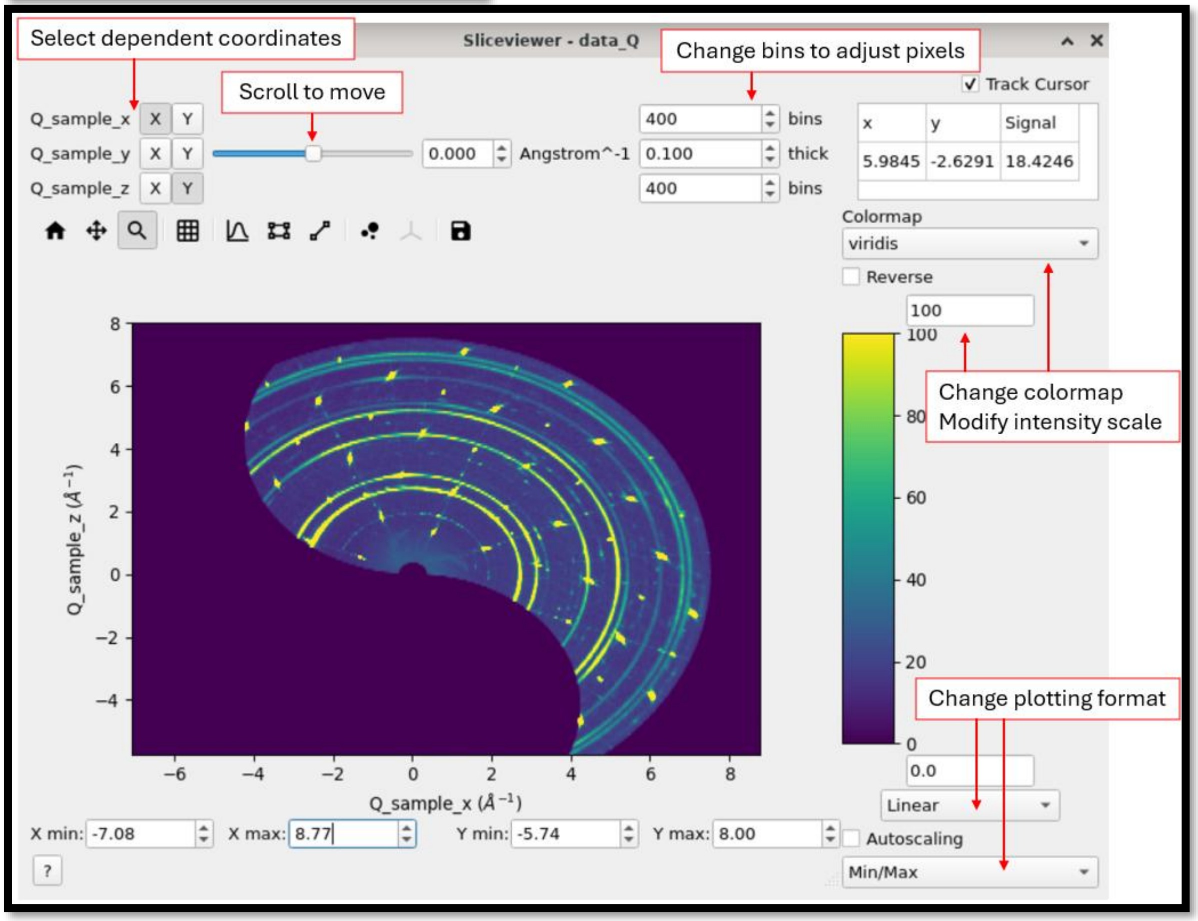
- *After it has finished running, a workspace **data_Q** appears in the "Workspaces" window on the left-hand side of the interface, which contains data in the reciprocal Q-space (**Q_sample**). Right-click on the workspace **data_Q** and select **Show Slice Viewer** to see what it looks like.*
- *Once the slice viewer window opens, there are several options to adjust the plot:*
    - *Select different dependent coordinates for 2D plotting.*

Back to Index

- *Scroll the 3rd coordinate to see different slicing.*
- *Adjust data bin size.*
- *Adjust intensity and colormap scales.*

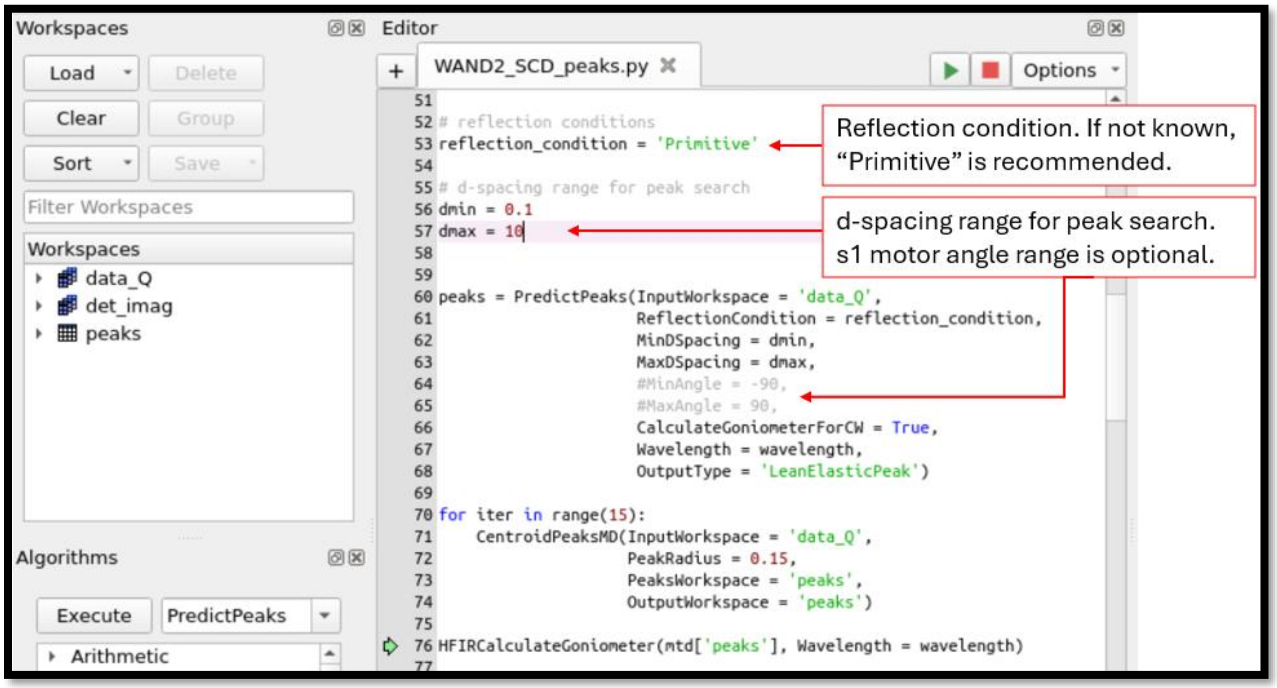


Note:
In the reciprocal Q-space (Q_sample), the goniometer tilting information is contained in the data. Usually, the sample is not perfectly aligned in the horizontal scattering plane, so we will see that the entire contour plot is "tilted" away from canonical xyz mesh grid.



Back to Index

(c) Predict and index peaks



```
51
52 # reflection conditions
53 reflection_condition = 'Primitive'
54
55 # d-spacing range for peak search
56 dmin = 0.1
57 dmax = 10
58
59
60 peaks = PredictPeaks(InputWorkspace = 'data_Q',
61                      ReflectionCondition = reflection_condition,
62                      MinDSpacing = dmin,
63                      MaxDSpacing = dmax,
64                      #MinAngle = -90,
65                      #MaxAngle = 90,
66                      CalculateGoniometerForCW = True,
67                      Wavelength = wavelength,
68                      OutputType = 'LeanElasticPeak')
69
70 for iter in range(15):
71     CentroidPeaksMD(InputWorkspace = 'data_Q',
72                     PeakRadius = 0.15,
73                     PeaksWorkspace = 'peaks',
74                     OutputWorkspace = 'peaks')
75
76 HFIRCalculateGoniometer(mtd['peaks'], Wavelength = wavelength)
```

Reflection condition. If not known, "Primitive" is recommended.

d-spacing range for peak search. s1 motor angle range is optional.

(1) As shown above, set **reflection_condition**. Available choices:

- "Primitive"
- "C-face centred"
- "A-face centred"
- "B-face centred"
- "Body centred"
- "All-face centred"
- "Rhombohedrally centred, obverse"
- "Rhombohedrally centred, reverse"
- "Hexagonally centred, reverse"

When not sure / want every possible peak, choose "Primitive".

(2) Set d-spacing range, **dmin** and **dmax**.
Or comment out "**MinDSpacing**" and "**MaxDSpacing**" lines, and use "**MinAngle**" and "**MaxAngle**" lines for peak search with a customized s1 motor angle range.

(3) Highlight command lines (line 53 to 76) shown below and hit **Run** (**green triangle**).

(4) A new workspace, **peaks**, appears in the "Workspaces" window. Right-click it and select **Show Data** to see its content → peak information summary (such as HKL index, wavelength, DSpacing, Qlab, Qsample).

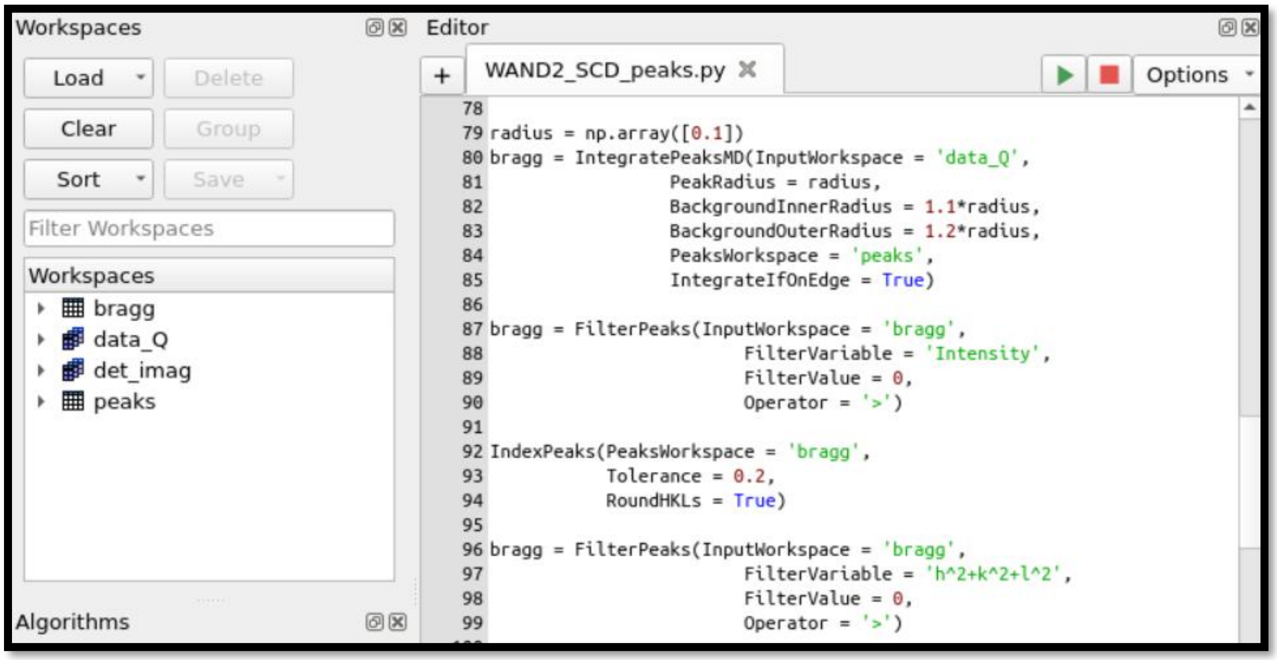
Back to Index

Note:

- ***CentroidPeaksMD** algorithm helps optimize the peak center ("center of intensity") for each peak in the Q-space.*
- ***HFIRCalculateGoniometer** algorithm recalculates the corresponding goniometer matrix for each peak with the incident neutron wavelength after peak center optimization.*
- *Peaks that are out of detectable range are also predicted. That's the reason we need to filter them out next.*

(d) Filter peaks (integer-HKL peaks)

(1) Set **radius** for parent peak filtering.
(2) Highlight command lines (line 79 to 99) shown below and hit the **Run** button (**green triangle**). A new workspace **bragg** will appear in the "Workspaces" window on the left.



```
78
79 radius = np.array([0.1])
80 bragg = IntegratePeaksMD(InputWorkspace = 'data_Q',
81                      PeakRadius = radius,
82                      BackgroundInnerRadius = 1.1*radius,
83                      BackgroundOuterRadius = 1.2*radius,
84                      PeaksWorkspace = 'peaks',
85                      IntegrateIfOnEdge = True)
86
87 bragg = FilterPeaks(InputWorkspace = 'bragg',
88                     FilterVariable = 'Intensity',
89                     FilterValue = 0,
90                     Operator = '>')
91
92 IndexPeaks(PeaksWorkspace = 'bragg',
93            Tolerance = 0.2,
94            RoundHKLs = True)
95
96 bragg = FilterPeaks(InputWorkspace = 'bragg',
97                     FilterVariable = 'h^2+k^2+l^2',
98                     FilterValue = 0,
99                     Operator = '>')
```

Note:

- *This part of command lines works as an intensity filter for parent peaks (integer-HKL indices).*

Back to Index

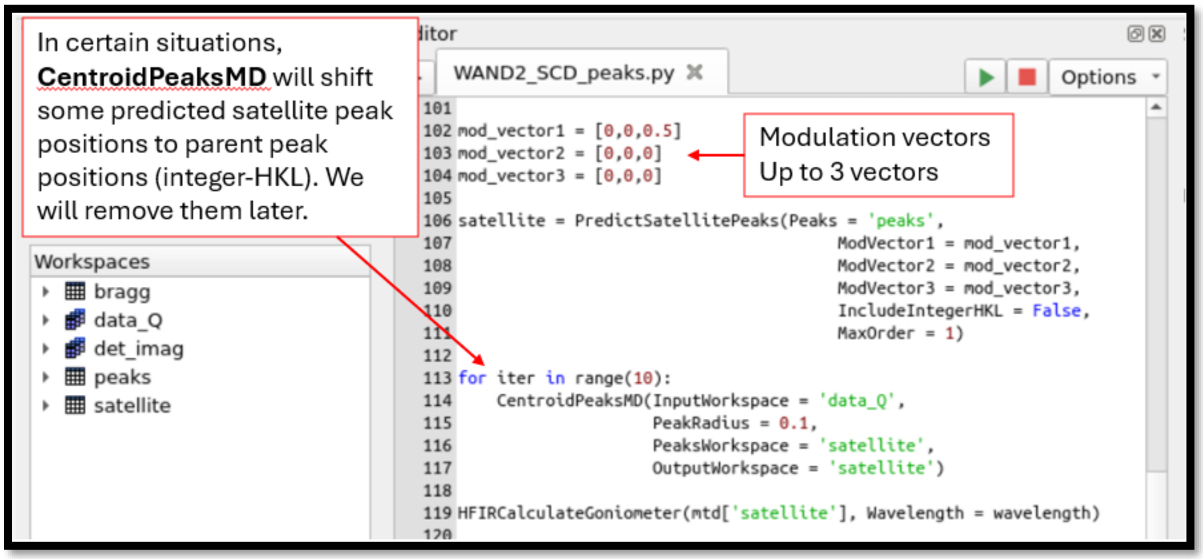
- ***IntegratePeaksMD*** *algorithm generates a sphere (or ellipsoid if a 3-number array is given to* ***radius****) centered at each peak with the preset radius (0.1 is sufficient. Unit:* Å^{-1}*) and integrates data within the sphere.* ***BackgroundInner/OuterRadius*** *is used for background estimation. The algorithm returns an estimation of peak intensities.*

- ***FilterPeaks*** *algorithm filters peaks based on the* ***FilterVariable*** *and* ***FilterValue****. For example,* ***FilterVariable = "Intensity"*** *with* ***FilterValue = 0*** *means only peaks with intensities > 0 are kept. This will remove those peaks that are out of detectable range.* ***FilterVariable = "Signal/Noise"*** *with* ***FilterValue = 1*** *is also commonly used to filter peaks.*

- ***IndexPeaks*** *algorithm will index the remaining peaks with H, K, L within the tolerance. If a peak cannot be indexed, it will have H, K, L = 0, 0, 0, and will be filtered out by the subsequent* ***FilterPeaks*** *with constraint of h^2+k^2+l^2 > 0.*

(e) Predict, index, and filter satellite peaks

(1) Set modulation vectors for satellite peaks.
(2) Highlight command lines (line 102 to 119) shown below and hit the **Run** button (**green triangle**).



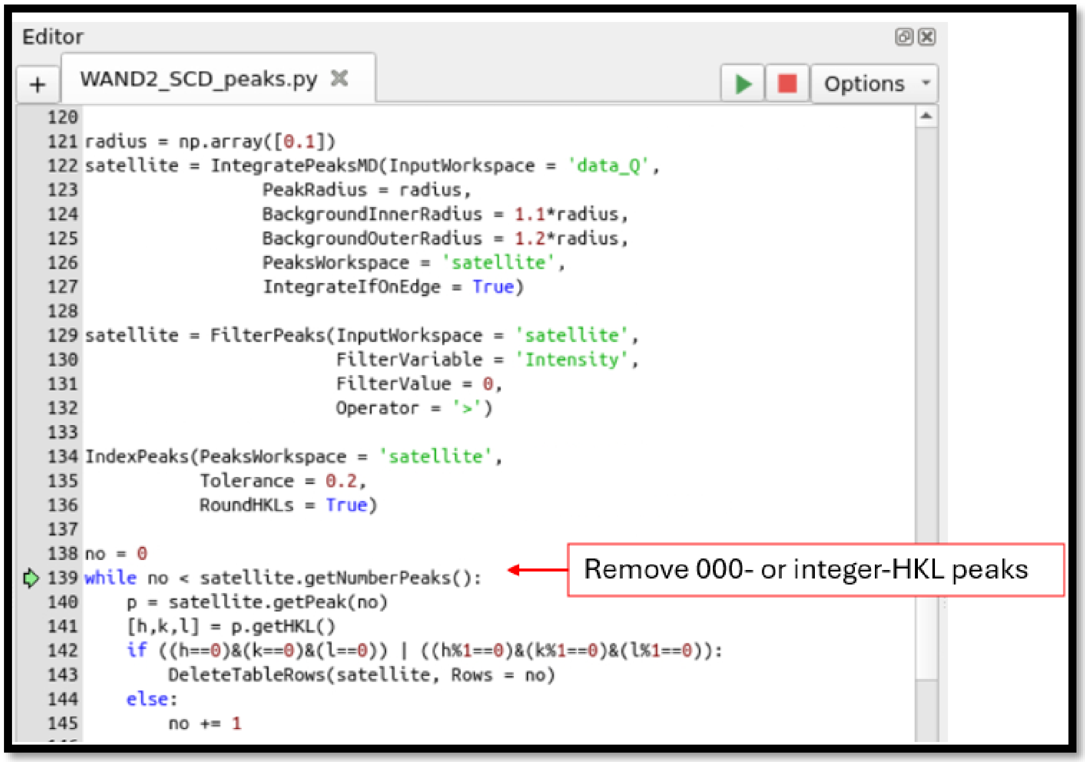
In certain situations, **CentroidPeaksMD** will shift some predicted satellite peak positions to parent peak positions (integer-HKL). We will remove them later.

Modulation vectors
Up to 3 vectors

```
101
102 mod_vector1 = [0,0,0.5]
103 mod_vector2 = [0,0,0]
104 mod_vector3 = [0,0,0]
105
106 satellite = PredictSatellitePeaks(Peaks = 'peaks',
107                                   ModVector1 = mod_vector1,
108                                   ModVector2 = mod_vector2,
109                                   ModVector3 = mod_vector3,
110                                   IncludeIntegerHKL = False,
111                                   MaxOrder = 1)
112
113 for iter in range(10):
114     CentroidPeaksMD(InputWorkspace = 'data_Q',
115                     PeakRadius = 0.1,
116                     PeaksWorkspace = 'satellite',
117                     OutputWorkspace = 'satellite')
118
119 HFIRCalculateGoniometer(mtd['satellite'], Wavelength = wavelength)
120
```

Back to Index

(3) A new workspace **satellite** will appear in the “Workspaces” window. Right-click it and select **Show Data** to see its content.
(4) Set **radius** for satellite peak filtering. Similar procedures as parent peak filtering.
(5) Highlight command lines (line 121 to 145) shown below and hit the **Run** button (**green triangle**).



```
120
121 radius = np.array([0.1])
122 satellite = IntegratePeaksMD(InputWorkspace = 'data_Q',
123                  PeakRadius = radius,
124                  BackgroundInnerRadius = 1.1*radius,
125                  BackgroundOuterRadius = 1.2*radius,
126                  PeaksWorkspace = 'satellite',
127                  IntegrateIfOnEdge = True)
128
129 satellite = FilterPeaks(InputWorkspace = 'satellite',
130                         FilterVariable = 'Intensity',
131                         FilterValue = 0,
132                         Operator = '>')
133
134 IndexPeaks(PeaksWorkspace = 'satellite',
135            Tolerance = 0.2,
136            RoundHKLs = True)
137
138 no = 0
139 while no < satellite.getNumberPeaks():
140     p = satellite.getPeak(no)
141     [h,k,l] = p.getHKL()
142     if ((h==0)&(k==0)&(l==0)) | ((h%1==0)&(k%1==0)&(l%1==0)):
143         DeleteTableRows(satellite, Rows = no)
144     else:
145         no += 1
```

Note:

- ***PredictSatellitePeaks** algorithm predicts satellite peaks based on the input modulation vectors and a parent peak workspace **peaks.***

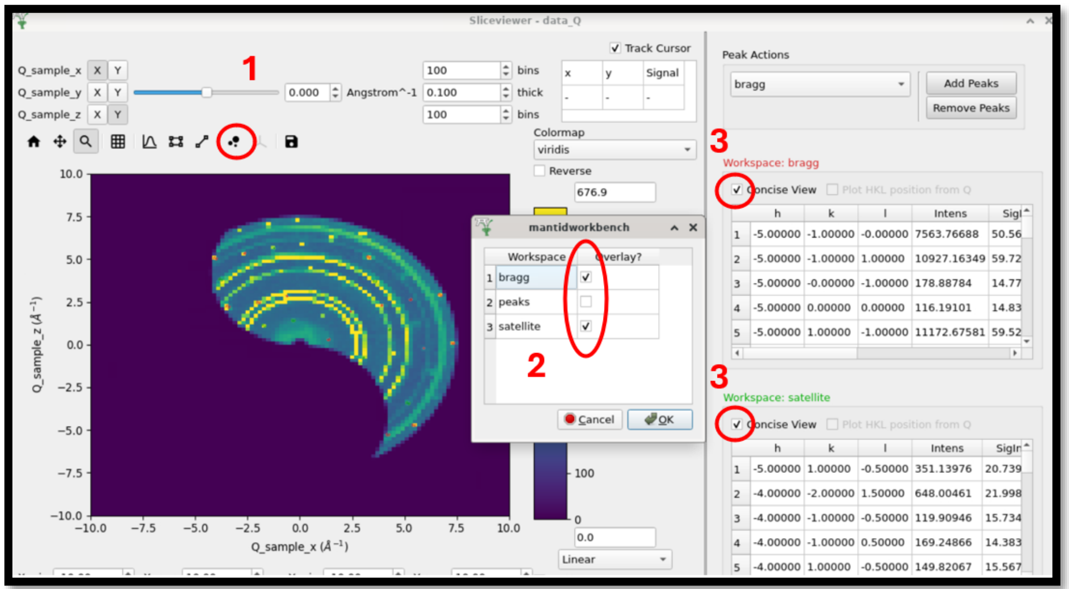
- *If there is no peak intensity at the predicted satellite peak position, **CentroidPeaksMD** may optimize them to the nearby parent peak position. That’s why we use line 139 to 145 to remove them from the satellite peak workspace.*

- *Line 139 to 145 will remove peaks indexed as (000) or integer-HKL in the workspace **satellite**.*

Back to Index

(f) Inspect peaks in Slice Viewer

(1) Right-click **data_Q** in the "Workspaces" window and select **Show Slice Viewer**.
(2) In the slice viewer window, click "Add peaks overlays" button (step **1**), shown below.
(3) Check the target peak workspace (step **2**) in the pop-up window and click OK. The workspace content will be listed on the right-hand side.
(4) Check "Concise View" (step **3**).
(5) Click on any peak in the right-hand side window (step **4**), for example (-5, -1, 1) here. The peak statistics will be highlighted in the left-hand side contour plot. The **red cross** in the plot is the peak center. Details will be covered in the later "Integrate peaks" section.



	Workspace	Overlay?
1	bragg	✓
2	peaks	
3	satellite	✓

bragg

Workspace: bragg

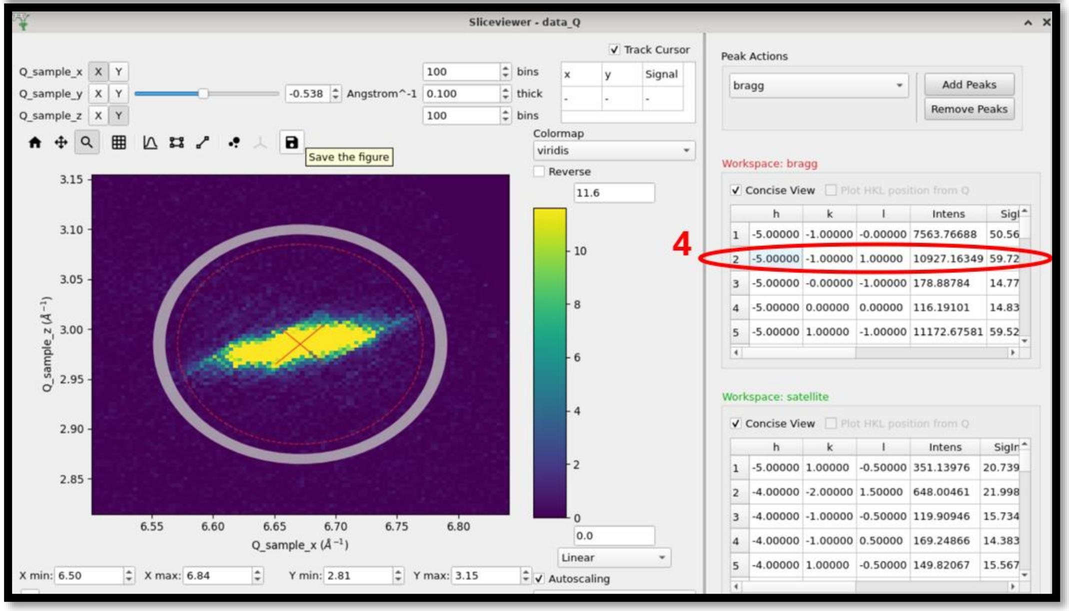
✓ Concise View ☐ Plot HKL position from Q

	h	k	l	Intens	Sigl
1	-5.00000	-1.00000	-0.00000	7563.76688	50.56
2	-5.00000	-1.00000	1.00000	10927.16349	59.72
3	-5.00000	-0.00000	-1.00000	178.88784	14.77
4	-5.00000	0.00000	0.00000	116.19101	14.83
5	-5.00000	1.00000	-1.00000	11172.67581	59.52

Workspace: satellite

✓ Concise View ☐ Plot HKL position from Q

	h	k	l	Intens	SigIn
1	-5.00000	1.00000	-0.50000	351.13976	20.739
2	-4.00000	-2.00000	1.50000	648.00461	21.998
3	-4.00000	-1.00000	-0.50000	119.90946	15.734
4	-4.00000	-1.00000	0.50000	169.24866	14.383
5	-4.00000	1.00000	-0.50000	149.82067	15.567



bragg

Workspace: bragg

✓ Concise View ☐ Plot HKL position from Q

	h	k	l	Intens	Sigl
1	-5.00000	-1.00000	-0.00000	7563.76688	50.56
2	-5.00000	-1.00000	1.00000	10927.16349	59.72
3	-5.00000	-0.00000	-1.00000	178.88784	14.77
4	-5.00000	0.00000	0.00000	116.19101	14.83
5	-5.00000	1.00000	-1.00000	11172.67581	59.52

Workspace: satellite

✓ Concise View ☐ Plot HKL position from Q

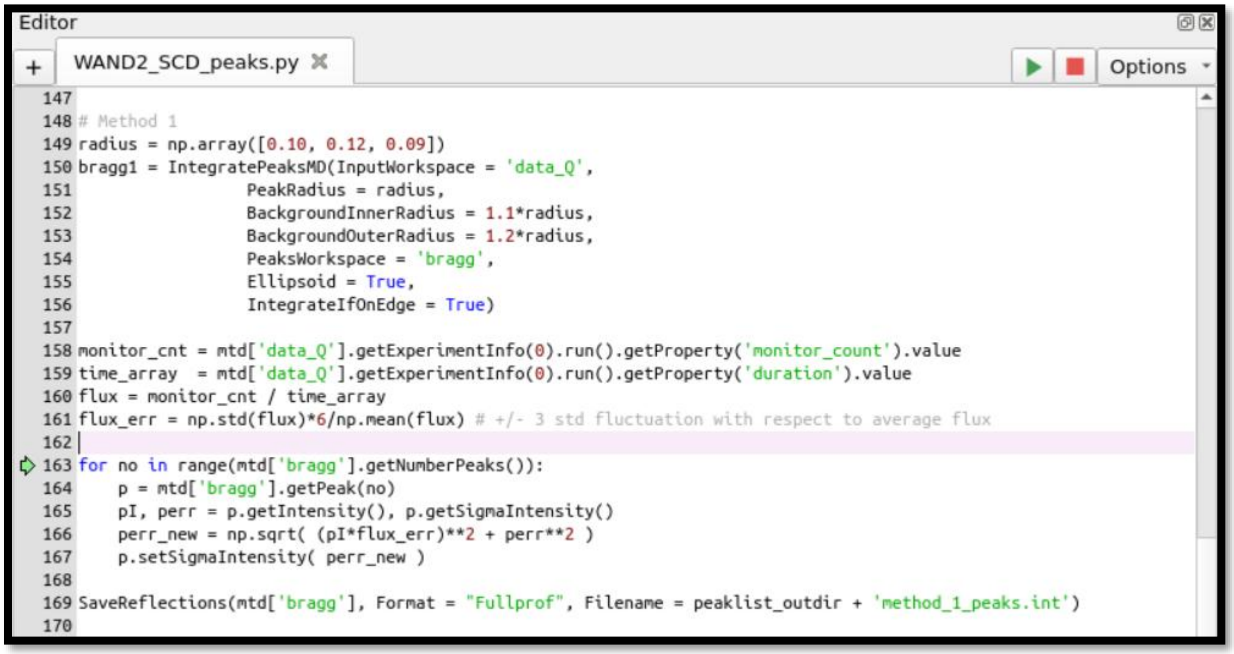
	h	k	l	Intens	SigIn
1	-5.00000	1.00000	-0.50000	351.13976	20.739
2	-4.00000	-2.00000	1.50000	648.00461	21.998
3	-4.00000	-1.00000	-0.50000	119.90946	15.734
4	-4.00000	-1.00000	0.50000	169.24866	14.383
5	-4.00000	1.00000	-0.50000	149.82067	15.567

Back to Index

(g) Integrate peaks

(i) Method 1: IntegratePeaksMD

(1) Set **radius** for peak integration in the command lines (line 149 to 156) shown below. In practice, a Bragg peak in Q-space does not have a perfect spherical profile. Instead of sphere, we use a 3D ellipsoid to do peak integration (as shown below, the command line "Ellipsoid = True" in the **IntegratePeaksMD** algorithm). The ellipsoid calls for 3 radii, which are semi-axis in Q_sample_x, Q_sample_y, Q_sample_z directions. Make sure the ellipsoid is big enough to cover the peak.
(2) Adjust **BackgroundInner/OuterRadius** coefficients. (Here are 1.1 and 1.2, respectively)
(3) Highlight these command lines (line 149 to 169) and hit **Run** (**green triangle**).
(4) Replace all **bragg** strings with **satellite** strings and change output file name when integrating satellite peaks in the same way.
(5) The for-loop (line 158 to 167) is to revise the errors of peak intensities, considering the uncertainty (%) of neutron beam flux.
(6) If the peak integration is not ideal, consider changing **radius** to re-integrate the peaks.



```
147
148 # Method 1
149 radius = np.array([0.10, 0.12, 0.09])
150 bragg1 = IntegratePeaksMD(InputWorkspace = 'data_Q',
151                      PeakRadius = radius,
152                      BackgroundInnerRadius = 1.1*radius,
153                      BackgroundOuterRadius = 1.2*radius,
154                      PeaksWorkspace = 'bragg',
155                      Ellipsoid = True,
156                      IntegrateIfOnEdge = True)
157
158 monitor_cnt = mtd['data_Q'].getExperimentInfo(0).run().getProperty('monitor_count').value
159 time_array  = mtd['data_Q'].getExperimentInfo(0).run().getProperty('duration').value
160 flux = monitor_cnt / time_array
161 flux_err = np.std(flux)*6/np.mean(flux) # +/- 3 std fluctuation with respect to average flux
162
163 for no in range(mtd['bragg'].getNumberPeaks()):
164     p = mtd['bragg'].getPeak(no)
165     pI, perr = p.getIntensity(), p.getSigmaIntensity()
166     perr_new = np.sqrt( (pI*flux_err)**2 + perr**2 )
167     p.setSigmaIntensity( perr_new )
168
169 SaveReflections(mtd['bragg'], Format = "Fullprof", Filename = peaklist_outdir + 'method_1_peaks.int')
170
```

Note:

- *A percentage uncertainty coming from neutron beam fluctuation is propagated into the estimation of peak intensity error. The percentage uncertainty is included in Method 2 as well.*

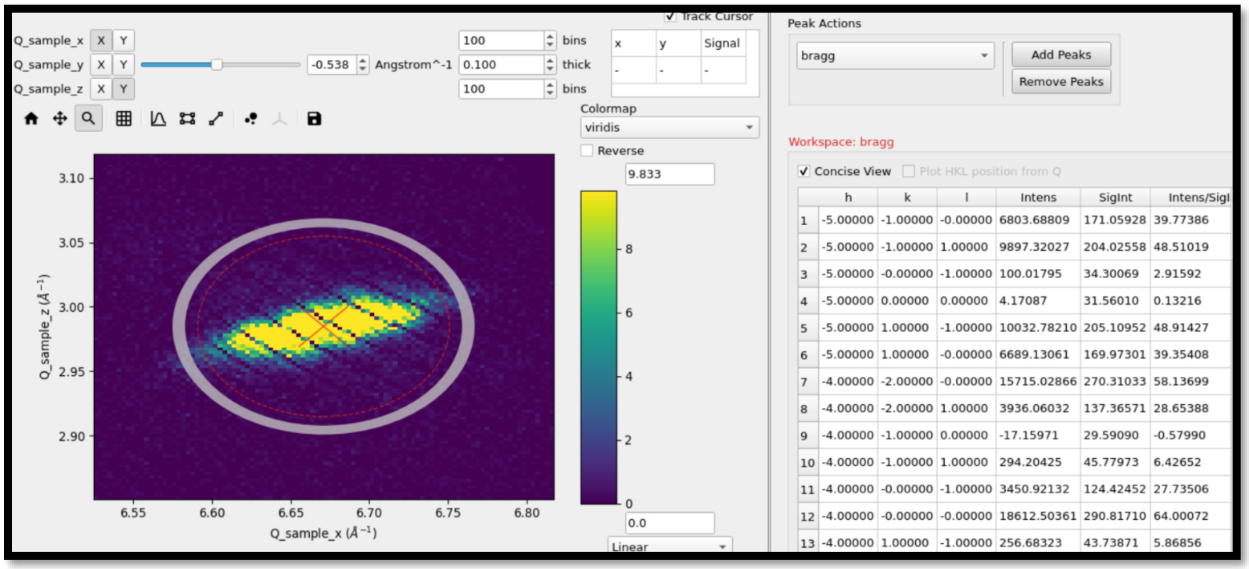
Back to Index

- ***SaveReflections*** *saves the peak workspace into a readable peak file for later refinement. Available formats:*
    - *"Fullprof"*
    - *"GSAS"*
    - *"Jana"*
    - *"SHELX"*

- *Details of **IntegratedPeaksMD** can be found on https://docs.mantidproject.org/nightly/algorithms/IntegratePeaksMD-v2.html.*

- *After execution, open **data_Q** in the slice viewer.*
    - *Add peak overlays as we do in the "Inspect peaks in Slice Viewer" section.*
    - *Click on any peak to check integration statistics. The integration ellipsoid will be visualized on the left-hand side, shown below.*
    - *The **red cross** is the peak center.*
    - *The **red dash ellipse** is the 3D ellipsoid range.*
    - *The **grey ring** is the region of background estimation, determined by BackgroundInner/OuterRadius.*

- *The error (**SigInt** shown in the table below) of the integrated peak intensity obtained from **IntegratedPeaksMD** is highly dependent on BackgroundInner/OuterRadius (try different settings to see the dependence). The percentage uncertainty is additionally propagated to the intensity error to alleviate the problem.*



	h	k	l	Intens	SigInt	Intens/SigI
1	-5.00000	-1.00000	-0.00000	6803.68809	171.05928	39.77386
2	-5.00000	-1.00000	1.00000	9897.32027	204.02558	48.51019
3	-5.00000	-0.00000	-1.00000	100.01795	34.30069	2.91592
4	-5.00000	0.00000	0.00000	4.17087	31.56010	0.13216
5	-5.00000	1.00000	-1.00000	10032.78210	205.10952	48.91427
6	-5.00000	1.00000	-0.00000	6689.13061	169.97301	39.35408
7	-4.00000	-2.00000	-0.00000	15715.02866	270.31033	58.13699
8	-4.00000	-2.00000	1.00000	3936.06032	137.36571	28.65388
9	-4.00000	-1.00000	0.00000	-17.15971	29.59090	-0.57990
10	-4.00000	-1.00000	1.00000	294.20425	45.77973	6.42652
11	-4.00000	-0.00000	-1.00000	3450.92132	124.42452	27.73506
12	-4.00000	-0.00000	-0.00000	18612.50361	290.81710	64.00072
13	-4.00000	1.00000	-1.00000	256.68323	43.73871	5.86856

[Back to Index](#)

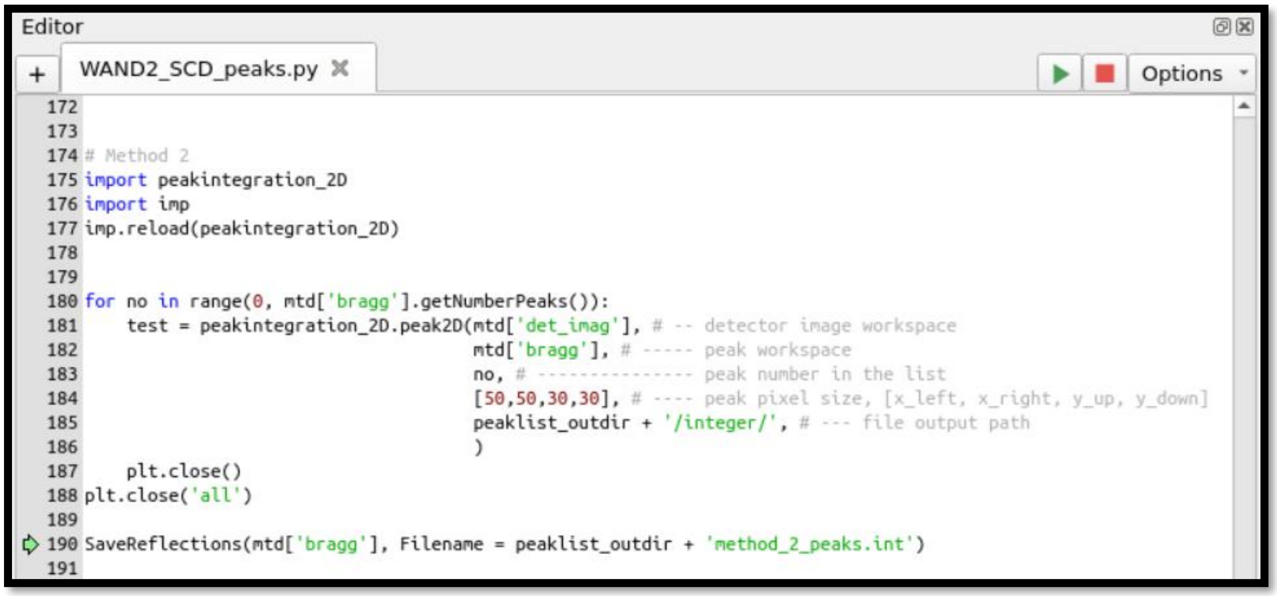
(ii) Method 2: Detector image integration

(1) Follow the path "/data/HFIR/shared/WANDscripts/" and locate the script

peakintegration_2D.py

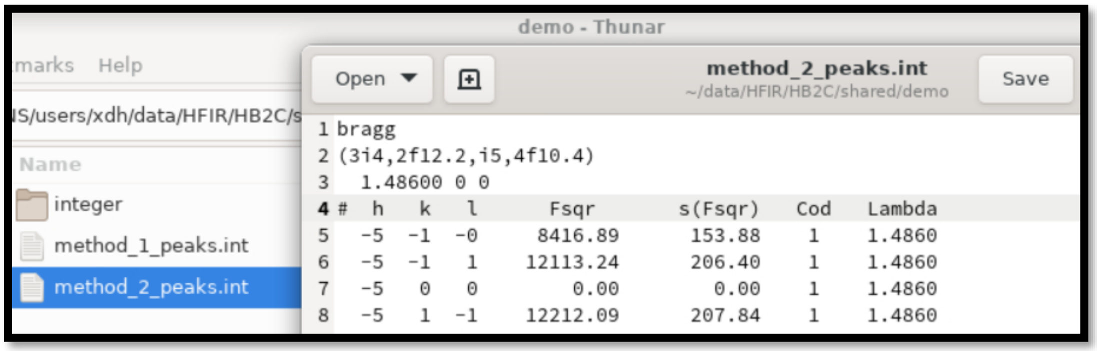
(2) Copy and paste this python file into your working directory.

(3) Highlight command lines (line 175 to 190) shown below and hit the **Run** button (**green triangle**).



```
172
173
174 # Method 2
175 import peakintegration_2D
176 import imp
177 imp.reload(peakintegration_2D)
178
179
180 for no in range(0, mtd['bragg'].getNumberPeaks()):
181     test = peakintegration_2D.peak2D(mtd['det_imag'], # -- detector image workspace
182                                      mtd['bragg'], # ----- peak workspace
183                                      no, # --------------- peak number in the list
184                                      [50,50,30,30], # ---- peak pixel size, [x_left, x_right, y_up, y_down]
185                                      peaklist_outdir + '/integer/', # --- file output path
186                                      )
187     plt.close()
188 plt.close('all')
189
190 SaveReflections(mtd['bragg'], Filename = peaklist_outdir + 'method_2_peaks.int')
191
```

(4) In the output directory, a corresponding folder containing png files of peak statistics and a peak list file (.int) are generated.



```
bragg
(3i4,2f12.2,i5,4f10.4)
  1.48600 0 0
#   h   k   l        Fsqr      s(Fsqr)   Cod    Lambda
   -5  -1  -0     8416.89       153.88     1    1.4860
   -5  -1   1    12113.24       206.40     1    1.4860
   -5   0   0        0.00         0.00     1    1.4860
   -5   1  -1    12212.09       207.84     1    1.4860
```

Note:

- *Parameter setting and explanation details can be found in the Appendix.*
- *To integrate satellite peaks, replace **bragg** strings with **satellite** strings, and replace the directory path and output file name with other distinguishable names as needed.*
- *In rare cases, if the integration fails for some peak(s), think about using **IntegratePeaksMD** (Method 1) to deal with it(them) specifically.*

Back to Index

Appendix

(a) Peak integration algorithm details (Method 2)

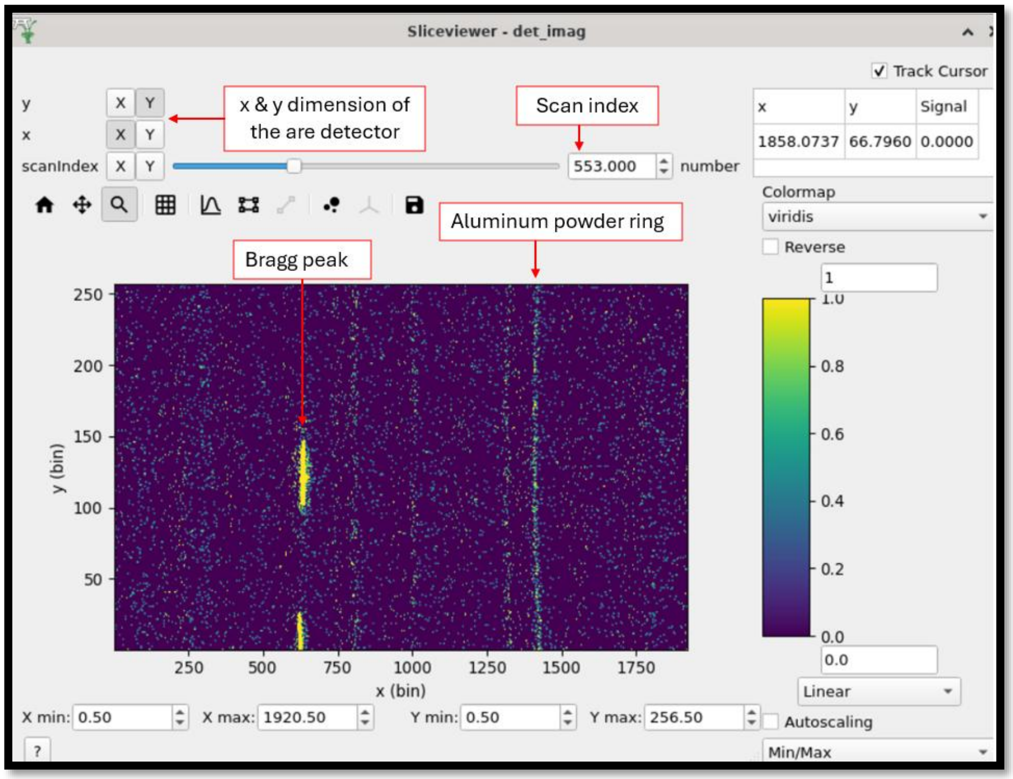
(1) **Data**

We first look at the data we use for peak integration --- detector image workspace. Right-click **det_imag** workspace and select **Show Slice Viewer**. A window containing the detector images will pop up (shown below).

- **scanIndex** is the number of the scan, which corresponds to an s1 motor angle value. If the measurement is a 180-degree rotation scan with a step size of 0.1 degree, there will be in total 180/0.1 = 1800 scans, and **scanIndex** will be 1~1800.

- **x** & **y** (or xbin & ybin) are dimensions of the area detector. There are in total 3840 x 512 = 1966080 detector pixels in the area detector, xbin = 1~3840, ybin = 1~512. Here we chose grouping = '2x2', so the regrouped xbin = 1~1920, ybin = 1~256.

- Bright spot in the 2D contour plot is a Bragg peak. Due to instrument resolution, the spot is elongated along the vertical direction (ybin). Aluminum powder ring can be seen as well.



x	y	Signal
1858.0737	66.7960	0.0000

scanIndex: 553.000

Colormap: viridis

X min: 0.50 X max: 1920.50 Y min: 0.50 Y max: 256.50

Back to Index

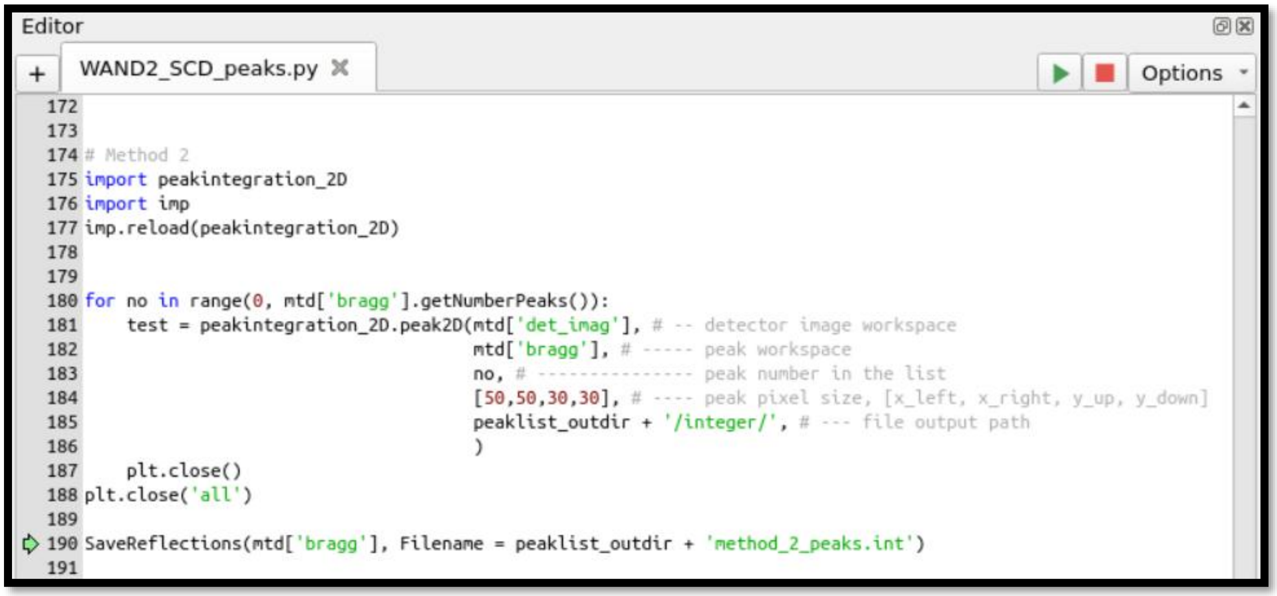
(2) **Algorithm input parameters**

As shown below, the algorithm calls for: (workspace names are free to change)

- **det_imag** ----- detector image workspace, generated during data loading
- **bragg** ----- target peak workspace
- **no** ----- peak number in the peak workspace, start from 0 (python gauge)
- **[x_left, x_right, y_up, y_down]** ----- 4-number array. It is the peak ROI size for integration (explained in "Algorithm output 2" later)
- **file output path** ----- a string describing the output directory path
- In **SaveReflections**, a file name is needed to store output peak information

Note:

- *"mtd" is a built-in function in Mantid. It assigns a handle to the workspace.*
- *"test" in line 181 means nothing, just a handle to execute algorithm.*



```
172
173
174 # Method 2
175 import peakintegration_2D
176 import imp
177 imp.reload(peakintegration_2D)
178
179
180 for no in range(0, mtd['bragg'].getNumberPeaks()):
181     test = peakintegration_2D.peak2D(mtd['det_imag'], # -- detector image workspace
182                                      mtd['bragg'], # ----- peak workspace
183                                      no, # --------------- peak number in the list
184                                      [50,50,30,30], # ---- peak pixel size, [x_left, x_right, y_up, y_down]
185                                      peaklist_outdir + '/integer/', # --- file output path
186                                      )
187     plt.close()
188 plt.close('all')
189
190 SaveReflections(mtd['bragg'], Filename = peaklist_outdir + 'method_2_peaks.int')
191
```

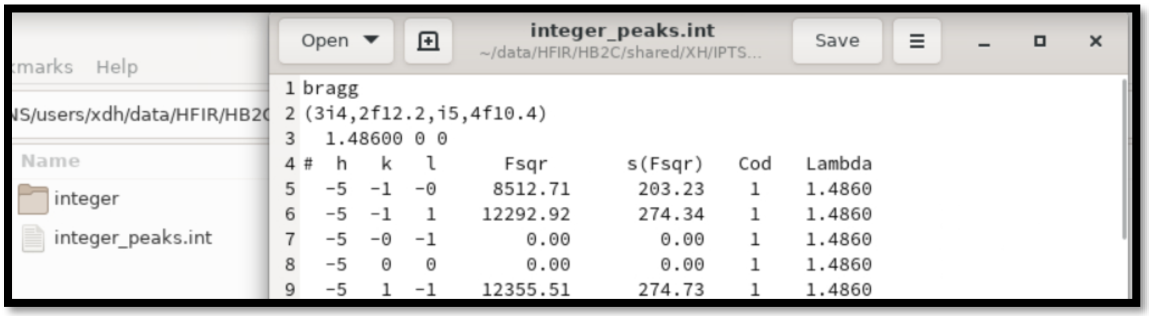
(3) **Algorithm output 1 --- peak list .int file**

SaveReflections (line 190) will save the target peak workspace into a peak list .int file in the output directory. The output file contains peak information, such as peak indices, intensity, and error for refinement (snapshot shown next for FullProf format).

Note:

Peak intensity = 0 means the peak is rejected. Remove it from the refinement.

Back to Index

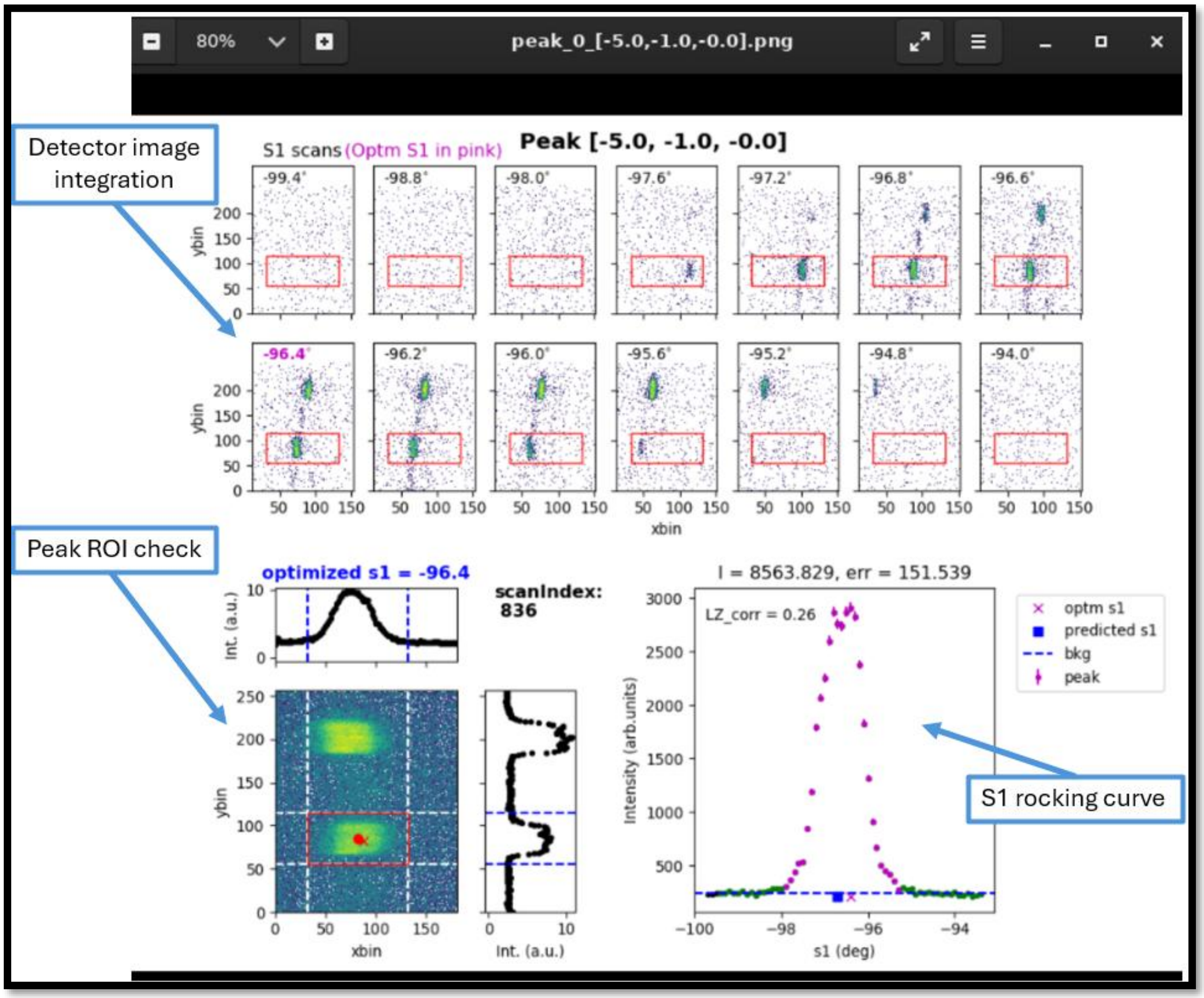


```
bragg
(3i4,2f12.2,i5,4f10.4)
  1.48600 0 0
#   h   k   l        Fsqr       s(Fsqr)   Cod   Lambda
   -5  -1  -0     8512.71      203.23     1    1.4860
   -5  -1   1    12292.92      274.34     1    1.4860
   -5  -0  -1        0.00        0.00     1    1.4860
   -5   0   0        0.00        0.00     1    1.4860
   -5   1  -1    12355.51      274.73     1    1.4860
```

(4) **Algorithm output 2 --- png files summarizing peak statistics**

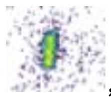
The algorithm will also generate a folder containing a series of png files, as shown above. Each png file contains integration information of each peak.

Below is a png file summarizing the Bragg peak (-5, 1, 0).

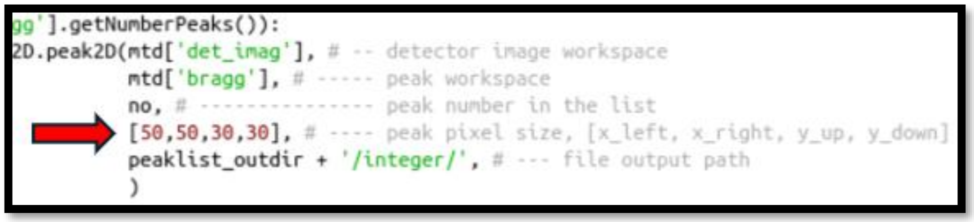


Back to Index

In this png file, the title shows the target peak (with index), and 3 sections are summarized:

- **Detector image integration:**
  - In the upper half of the png file, a series of detector images are listed. Each image corresponds to an s1 motor value (the numerical value shown in each image).
  - In each detector image, one colored spot, something like this , corresponds to one peak in the logarithmic color scale. A peak will gradually appear in the detector image and then fade away as s1 motor rotates, as observed in the example above for peak (-5, -1, 0). The image with **bold magenta** s1 value refers to where peak intensity collected in the detector image reaches its maximum as s1 motor rotates.
  - The **red box** in each image highlights the integration region. As long as the peak intensity is covered by the **red box**, the integration is legit. The size of the **red box** is determined by the input 4-number array (see in "Peak ROI check" next).
  - If multiple peaks appear in the **red box** of the same detector image, be careful. It could be due to twin domains, or impurity, or very close satellite peaks. Think about dealing with it using **IntegratePeaksMD** (Method 1) if necessary.

- **Peak ROI (region-of-interest) check**:
  - This panel shows the part of detector image around the target peak. The **red cross** is the predicted peak center. The **red dot** is the optimized peak center. If they differ from each other substantially (e.g., half of the **red box** size), consider optimizing the peak prediction, and redo the integration.



```
gg'].getNumberPeaks()):
2D.peak2D(mtd['det_imag'], # -- detector image workspace
          mtd['bragg'], # ----- peak workspace
          no, # ---------------- peak number in the list
          [50,50,30,30], # ---- peak pixel size, [x_left, x_right, y_up, y_down]
          peaklist_outdir + '/integer/', # --- file output path
          )
```

  - The size of the **red box** in this contour plot is determined by the input 4-number array (shown above). x_left and x_right determine the box size

Back to Index

along xbin direction with respect to the optimized peak center (**red dot**). y_up and y_down determine the size along ybin direction. If the target peak falls outside the **red box**, modify this 4-number array accordingly.

- This **red box** is the same **red box** shown in the upper series of detector images.

- The panels on top and right-hand side of the contour plot are ybin-averaged and xbin-averaged cuts of the contour plot, respectively. The **blue dash lines** align with the **red box** boundary in the contour plot. They help users decide whether the **red box** size is proper to cover the target peak.

- **S1 rocking curve**:
  - The data in the **red box** of each detector image (shown in the upper half of the png file) are integrated. The integrated intensities are plotted as a function of the s1 motor value, shown in this panel.
  - **Blue dashed line** is the Gaussian-fitted background level.
  - **Purple dots** are the integrated peak range.
  - The integrated peak intensity is calculated by integrating the **purple dots** and subtracting the background level.
  - LZ_corr refers to a correction coefficient --- Lorentzian correction, calculated using peak positions. Information of Lorentzian correction can be found at https://single-crystal.ornl.gov/more/integration/index.html.
  - Final integrated peak intensity and its error are shown in the panel title.

Note:

- *The **red box** (ROI) can help distinguish the target peak from other adjacent peaks.*
- *If the target peak intensity is fully contained in the **red box** (ROI), the peak profile (shape) will not affect the integration.*
- *When the target peak is close to or sits on top of the aluminum powder ring, the powder ring signal would be contained in the **red box**. This contained part of powder ring signal can be approximated to be constant as s1 changes within the range of +/- 2~3 degrees. Thus, the powder ring signal will be part of the background level in the s1 rocking curve and hence be filtered out from the final integrated peak intensity.*

Back to Index

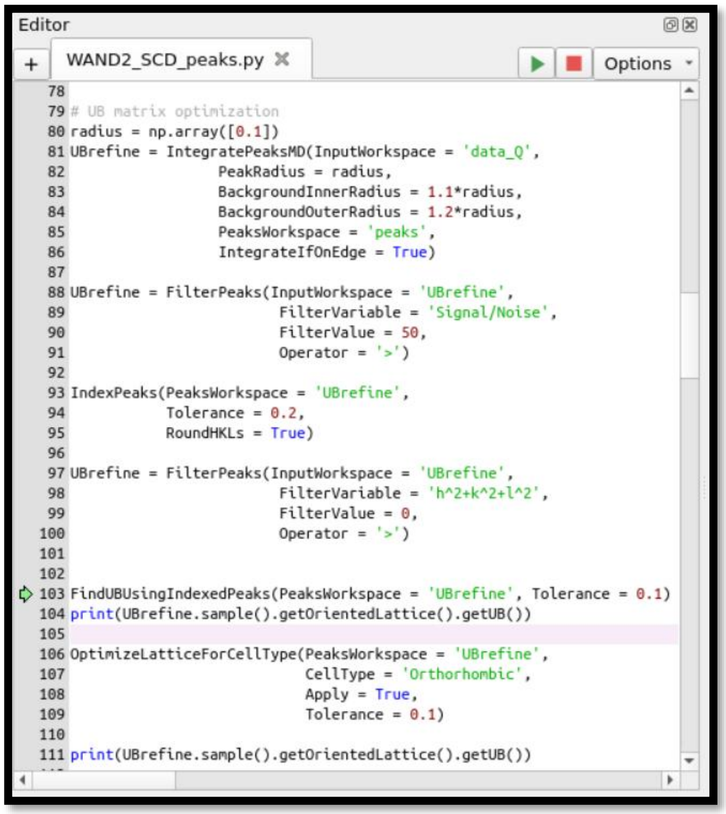
(b) UB matrix optimization

In certain situations, we want to optimize the UB matrix. A straightforward way is to refine UB matrix using strong Bragg peaks (integer-HKL peaks). To do so, after the peak prediction, we add/use and execute command lines shown below:

(1) Set radius for peak filtering. We use **IntegratedPeaksMD** to integrate peaks with assigned spherical radius → we get a peak workspace **UBrefine**.
(2) Filter peaks based on Signal/Noise ratio. Here we set the threshold = 50. Larger threshold value gives us stronger peaks remaining. Modify it accordingly.
(3) **IndexPeaks** will index peaks in **UBrefine**.
(4) **FilterPeaks** will filter out peaks that cannot be indexed (HKL = (0,0,0)).
(5) Two options for optimizing the UB matrix:

- **FindUBUsingIndexedPeaks** uses peak information (e.g., Q-positions) from **UBrefine** to calculate an optimized UB matrix within assigned tolerance.

- **OptimizeLatticeForCellType** calculates an optimized UB matrix with the constraint of "**CellType**". Available choices for cell type: "Cubic", "Tetragonal", "Orthorhombic", "Hexagonal", Rhombohedral", "Monoclinic", "Triclinic".


Editor
WAND2_SCD_peaks.py

```

# UB matrix optimization
radius = np.array([0.1])
UBrefine = IntegratePeaksMD(InputWorkspace = 'data_Q',
                  PeakRadius = radius,
                  BackgroundInnerRadius = 1.1*radius,
                  BackgroundOuterRadius = 1.2*radius,
                  PeaksWorkspace = 'peaks',
                  IntegrateIfOnEdge = True)

UBrefine = FilterPeaks(InputWorkspace = 'UBrefine',
                       FilterVariable = 'Signal/Noise',
                       FilterValue = 50,
                       Operator = '>')

IndexPeaks(PeaksWorkspace = 'UBrefine',
           Tolerance = 0.2,
           RoundHKLs = True)

UBrefine = FilterPeaks(InputWorkspace = 'UBrefine',
                       FilterVariable = 'h^2+k^2+l^2',
                       FilterValue = 0,
                       Operator = '>')


FindUBUsingIndexedPeaks(PeaksWorkspace = 'UBrefine', Tolerance = 0.1)
print(UBrefine.sample().getOrientedLattice().getUB())

OptimizeLatticeForCellType(PeaksWorkspace = 'UBrefine',
                           CellType = 'Orthorhombic',
                           Apply = True,
                           Tolerance = 0.1)

print(UBrefine.sample().getOrientedLattice().getUB())
```

Note:

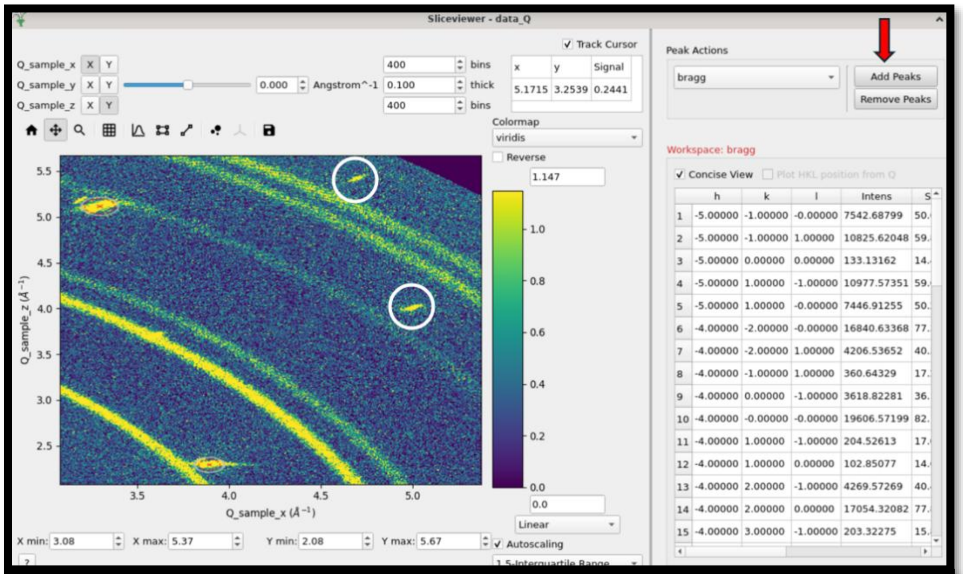
- *Both methods will update the UB matrix stored in the peak workspace **UBrefine**.*
- *The command line **print(UBrefine.sample().getOrientedLattice().getUB())** will print the UB matrix in the "Message" window on the right-hand side of the interface.*
- *Use **SetUB** algorithm to update the UB matrix of data workspace if needed (https://docs.mantidproject.org/nightly/algorithms/SetUB-v1.html)*

Back to Index

(c) Add, remove, adjust peaks for peak analysis

(i) Add peaks

In some rare situations, some of the observed peaks are not included in the peak workspace. For example, in the slice viewer shown below, each peak in the **bragg** workspace will have a **red cross** assigned at its peak position. However, we see that there are two peaks not included in the **bragg** workspace (highlighted by white circles).



Q_sample_y: 0.000 Angstrom^-1; bins 400, thick 0.100, bins 400

x	y	Signal
5.1715	3.2539	0.2441

Peak Actions: bragg

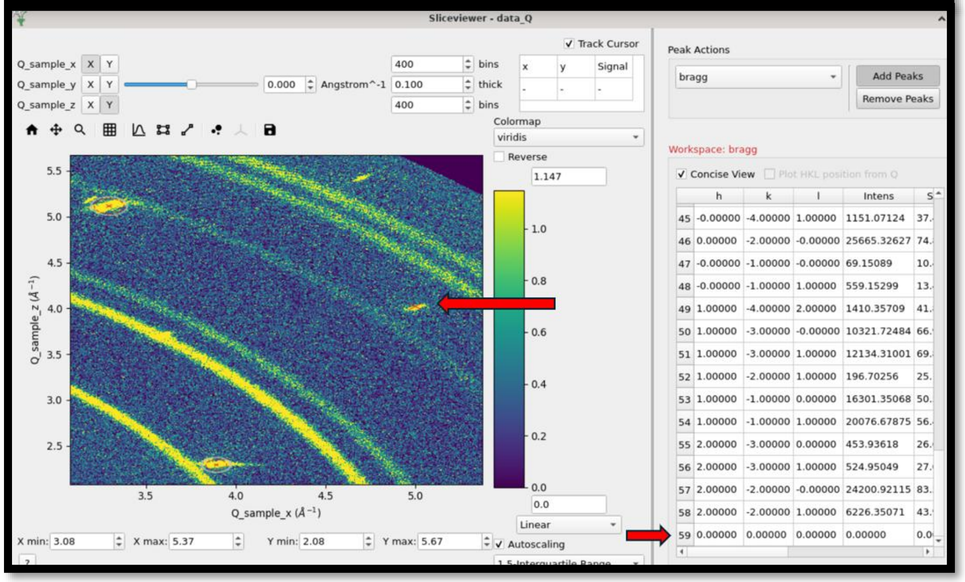
Colormap: viridis; 1.147; 0.0

X min: 3.08 X max: 5.37 Y min: 2.08 Y max: 5.67

Workspace: bragg

✓ Concise View ☐ Plot HKL position from Q

	h	k	l	Intens	S
1	-5.00000	-1.00000	-0.00000	7542.68799	50.
2	-5.00000	-1.00000	1.00000	10825.62048	59.
3	-5.00000	0.00000	0.00000	133.13162	14.
4	-5.00000	1.00000	-1.00000	10977.57351	59.
5	-5.00000	1.00000	-0.00000	7446.91255	50.
6	-4.00000	-2.00000	-0.00000	16840.63368	77.
7	-4.00000	-2.00000	1.00000	4206.53652	40.
8	-4.00000	-1.00000	1.00000	360.64329	17.
9	-4.00000	0.00000	-1.00000	3618.82281	36.
10	-4.00000	-0.00000	-0.00000	19606.57199	82.
11	-4.00000	1.00000	-1.00000	204.52613	17.
12	-4.00000	1.00000	0.00000	102.85077	14.
13	-4.00000	2.00000	-1.00000	4269.57269	40.
14	-4.00000	2.00000	0.00000	17054.32082	77.
15	-4.00000	3.00000	-1.00000	203.32275	15.



Q_sample_y: 0.000 Angstrom^-1; bins 400, thick 0.100, bins 400

x	y	Signal
-	-	-

Peak Actions: bragg

Colormap: viridis; 1.147; 0.0

X min: 3.08 X max: 5.37 Y min: 2.08 Y max: 5.67

Workspace: bragg

✓ Concise View ☐ Plot HKL position from Q

	h	k	l	Intens	S
45	-0.00000	-4.00000	1.00000	1151.07124	37.
46	0.00000	-2.00000	-0.00000	25665.32627	74.
47	-0.00000	-1.00000	-0.00000	69.15089	10.
48	-0.00000	-1.00000	1.00000	559.15299	13.
49	1.00000	-4.00000	2.00000	1410.35709	41.
50	1.00000	-3.00000	-0.00000	10321.72484	66.
51	1.00000	-3.00000	1.00000	12134.31001	69.
52	1.00000	-2.00000	1.00000	196.70256	25.
53	1.00000	-1.00000	0.00000	16301.35068	50.
54	1.00000	-1.00000	1.00000	20076.67875	56.
55	2.00000	-3.00000	0.00000	453.93618	26.
56	2.00000	-3.00000	1.00000	524.95049	27.
57	2.00000	-2.00000	-0.00000	24200.92115	83.
58	2.00000	-2.00000	1.00000	6226.35071	43.
59	0.00000	0.00000	0.00000	0.00000	0.0

To add these two peaks to the peak workspace,

(1) Click on **Add Peaks** button on the upper right corner.

(2) Move the mouse to the peak position and click. A **red cross** will appear at the peak position, and the new peak will be included at the end of the workspace table (right-hand side) with a (0,0,0) index.

(3) Run peak indexing command lines again.

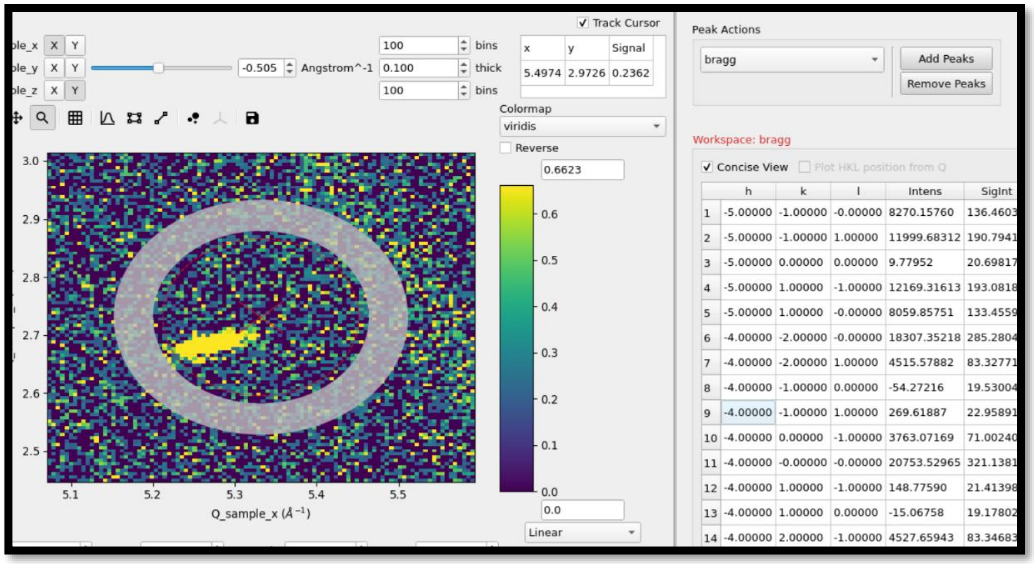
Back to Index

(i) Remove/Adjust peaks

In more common situations, some of the peaks are not well predicted, as shown below. For peak (-4, -1, 1), its predicted peak position (**red cross**) clearly deviates from the peak intensity spot. To adjust the prediction, follow the path:

- Click "Remove Peaks" on the upper right → Click on the **red cross** in the color plot (this will remove the predicted peak from the peak workspace) → Click "Add Peaks" on the upper right → Click at the expected peak intensity spot (a new **red cross** appears at the expected position, shown below).

This operation will include the target peak in the peak workspace with a (0,0,0) index with the revised peak position, shown in the second snapshot. Run peak indexing command lines again to revise the peak workspace.

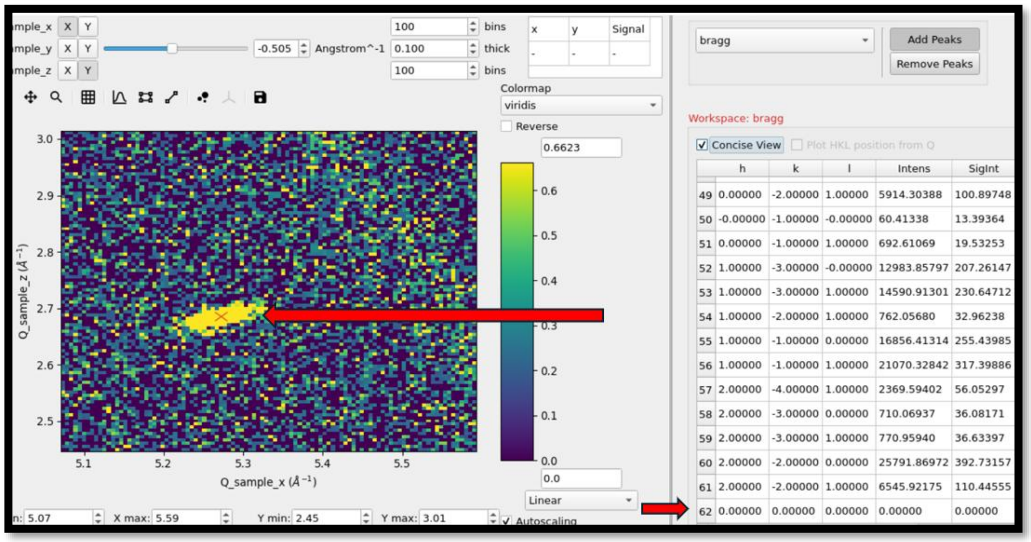


x	y	Signal
5.4974	2.9726	0.2362

Workspace: bragg

✓ Concise View ☐ Plot HKL position from Q

	h	k	l	Intens	SigInt
1	-5.00000	-1.00000	-0.00000	8270.15760	136.4603
2	-5.00000	-1.00000	1.00000	11999.68312	190.7941
3	-5.00000	0.00000	0.00000	9.77952	20.69817
4	-5.00000	1.00000	-1.00000	12169.31613	193.0818
5	-5.00000	1.00000	-0.00000	8059.85751	133.4559
6	-4.00000	-2.00000	-0.00000	18307.35218	285.2804
7	-4.00000	-2.00000	1.00000	4515.57882	83.32771
8	-4.00000	-1.00000	0.00000	-54.27216	19.53004
9	-4.00000	-1.00000	1.00000	269.61887	22.95891
10	-4.00000	0.00000	-1.00000	3763.07169	71.00240
11	-4.00000	-0.00000	-0.00000	20753.52965	321.1381
12	-4.00000	1.00000	-1.00000	148.77590	21.41398
13	-4.00000	1.00000	0.00000	-15.06758	19.17802
14	-4.00000	2.00000	-1.00000	4527.65943	83.34683



Workspace: bragg

✓ Concise View ☐ Plot HKL position from Q

	h	k	l	Intens	SigInt
49	0.00000	-2.00000	1.00000	5914.30388	100.89748
50	-0.00000	-1.00000	-0.00000	60.41338	13.39364
51	0.00000	-1.00000	1.00000	692.61069	19.53253
52	1.00000	-3.00000	-0.00000	12983.85797	207.26147
53	1.00000	-3.00000	1.00000	14590.91301	230.64712
54	1.00000	-2.00000	1.00000	762.05680	32.96238
55	1.00000	-1.00000	0.00000	16856.41314	255.43985
56	1.00000	-1.00000	1.00000	21070.32842	317.39886
57	2.00000	-4.00000	1.00000	2369.59402	56.05297
58	2.00000	-3.00000	0.00000	710.06937	36.08171
59	2.00000	-3.00000	1.00000	770.95940	36.63397
60	2.00000	-2.00000	0.00000	25791.86972	392.73157
61	2.00000	-2.00000	1.00000	6545.92175	110.44555
62	0.00000	0.00000	0.00000	0.00000	0.00000

Back to Index