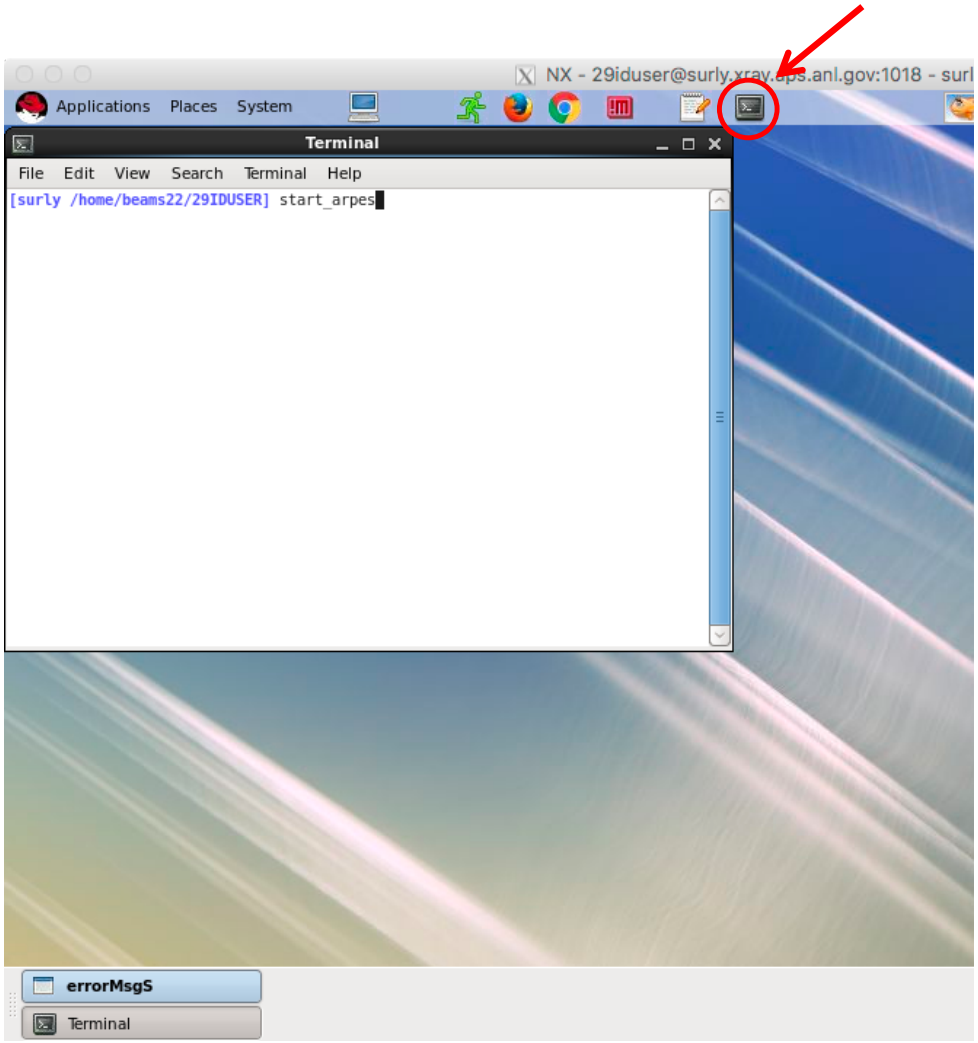
29ID
User Manual

Start-up

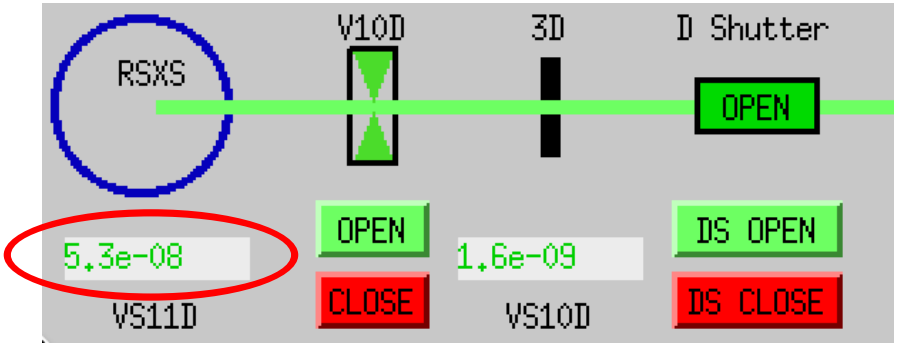
- Log-on into any linux box:
  - User: 29iduser
  - Pwd: *ask staff*
- **ARPES (29ID-C)**:
  - Open a terminal
  - type: *start_arpes*
  - 3 screens pop-up:
    - Beamline overview (BL_User)
    - ARPES manipulator (29idc_graphic)
    - Electron Analyzer (29idc_SESall)
- **RSXS (29ID-D)**:
  - Open a terminal:
  - type: *start_rsxs*
  - 2 screens pop-up:
    - Beamline overview (BL_User)
    - RSXS manipulator (29idd_graphic)
- Ask beamline staff to set-up a user folder.



!!! WARNINGS !!!

Before <u>transferring</u> a sample or when you are done for the day, always :

(1) turn off detector HV
(2) close the C/D-shutter
(3) close the chamber valve.



Pressure in the endstation VS11C (D) needs < $1e^{-9}$ ($1e^{-7}$) Torr to open to the beamline and turn on detector HV

ARPES endstation:

- Mirror chamber ~ 10^{-10} Torr (upstream of chamber valve)
- Main chamber IG~ 10^{-10} Torr
- Intro chamber IP ~ 10^{-10} Torr
- Load Lock ~ 10^{-7} Torr
- MCP/Screen HV: 1550 / 3600 V

RSXS endstation :

- Mirror chamber ~ 10^{-9} Torr (upstream of chamber valve)
- Main chamber ~ 10^{-8} Torr
- Load Lock ~ 10^{-6} Torr
- MCP HV: -2570 V

Beamline Overview (1)

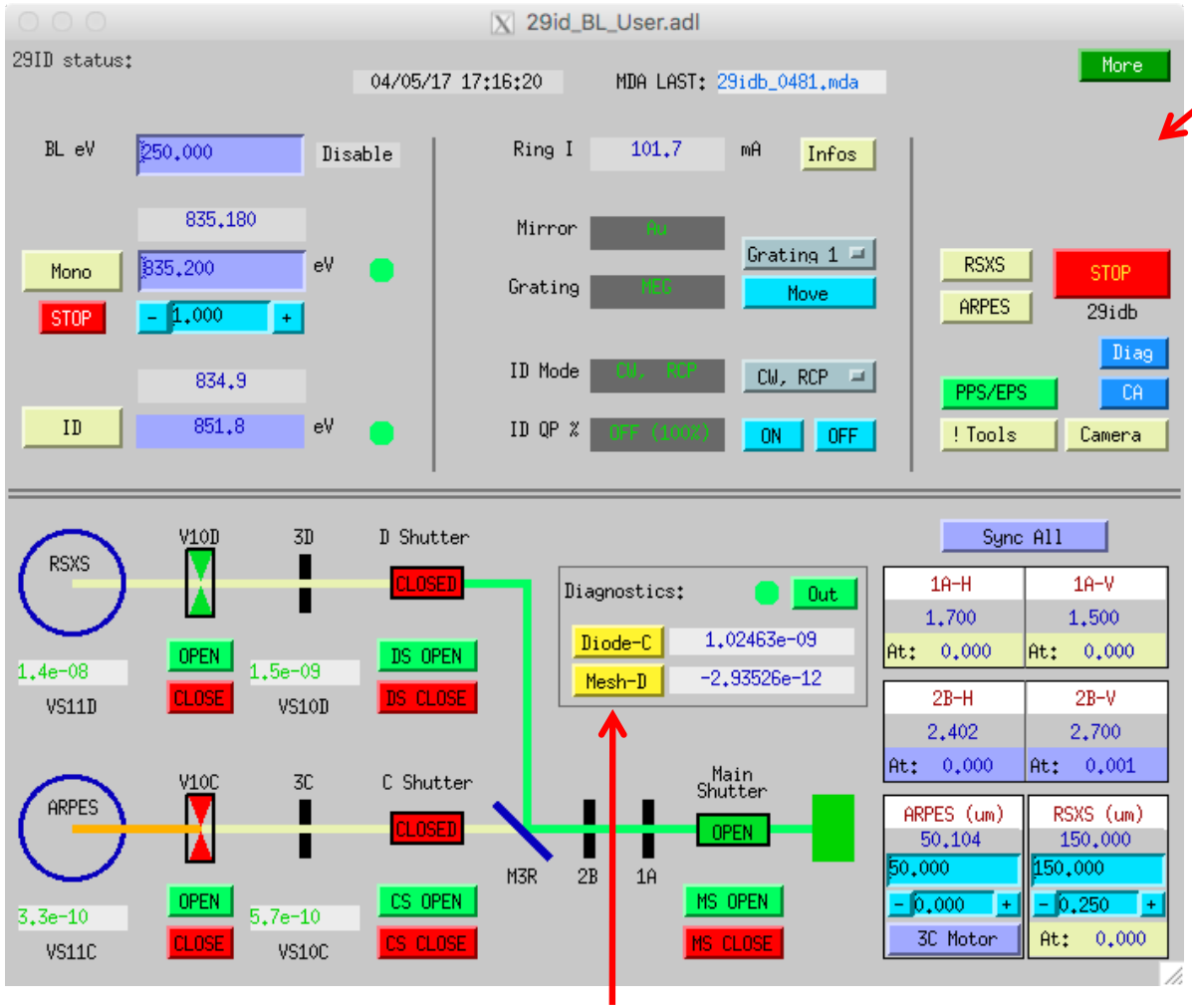
Energy (p5)

Beam characteristics (p5)

Motors Scan/tools (p6)

Emergency STOP ALL beamline motors

Min Energy:
H: 250 eV
V: 440 eV
CP: 400 eV
Max Energy:
MEG: 3000 eV
HEG: 2000 eV



Beamline apertures

ARPES/RSXS energy resolving slits (μm)

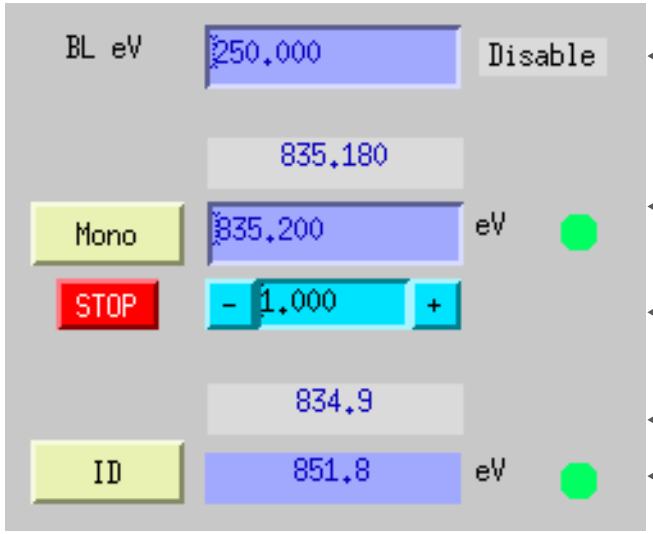
Beamline status:

- X-ray
- Shutter closed
- Valve(s) closed

There are many valves along the beamline (not shown here).
If the BL turns orange with the chamber valve open, contact staff.

Drop in/out diagnostic

Beamline Overview (2)



BL energy: drives mono, ID & slits 1A/2B
Must be in "enable" position

Only used for energy scans (see python scripts)

Mono readback (RBV, top) and set point (SP, bottom)

Tweak button: for mono only; use to optimize flux
STOP: stops the mono only

ID readback: can be off by a few %
ID Set Point (use Python script SetBL(energy))

Status buttons:

- ready
- busy
- ID - no permission
- ID - off

IMPORTANT:

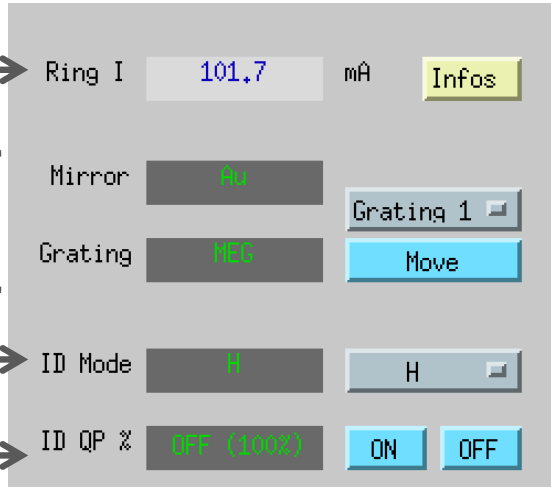
- The ID can take **up to 6 minutes** to go around its hysteresis loop – this is **normal**.
- If the ID stays in the busy state for more than 10 minutes, you can restart it either by using the Python script *IDRestart ()* or changing mode
- If ID is off: use Python script *IDStart(mode)* with mode = "H", "V", "RCP", "LCP"
- The mono is fast – if it stays "busy" for more than a couple of minutes, call staff

Storage Ring current & info (check for beam dump update)

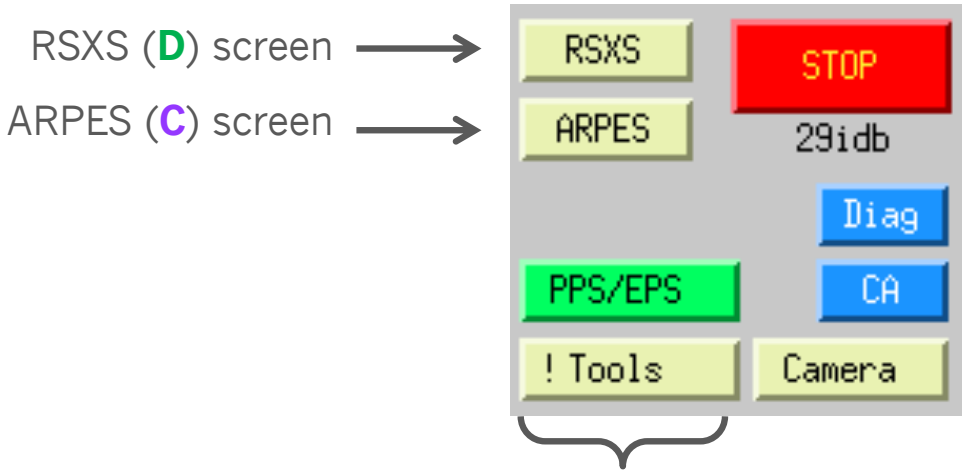
Mono optical components

ID polarization (H, V, RCP, LCP)

Quasiperiodicity Ask staff before turning ON



Beamline Overview (3)



RSXS (**D**) screen

ARPES (**C**) screen

Emergency stop:
stops ALL beamline motors up to the end-station valve;
Stop for endstation motor located on ARPES/RSXS screens

Diag: diagnostic screen
CA: current amplifier screen

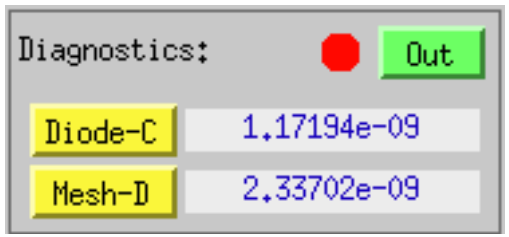
Cameras IOC (ask staff)

Data acquisition tools:

- dview : data viewer (p10)
- sview : live scan viewer
- striptool : pv monitoring (p11)
- ImageJ : cameras Image
- iPython : BL scripts (p7)

Red = diagnostic potentially blocking the beam

Clear all diagnostics



Insert diode in C branch

Insert mesh in D branch

Current on mesh/diode (ca14/ca15)
Use for normalization (I_0)

iPython - Beamline

When starting iPython from `! Tools`, all of the BL functions are automatically loaded & compiled:

```
In [1]: SetBL(1125)              # sets BL energy (ID & Mono) and corresponding apertures (1A & 2B)

In [1]: SetMono(1125)            # sets Mono energy only

In [1]: SetExitSlit (100)        # sets Exit Slit (μm) :  ➪ ARPES:   5 < Exit Slit < 300 μm
                                                        ➪ RSXS:    5 < Exit Slit < 500 μm

In [1]: Switch_IDMode ("X")      # Switch ID Polarization   – with "X" = "RCP", "LCP", "H", "V"

In [1]: Switch_Grating ("X")     # Switch Mono Grating      – with "X" = "HEG", "MEG"

In [1]: IDStart ("X")            # Turn on the ID (see p5) – with "X" = "RCP", "LCP", "H", "V"

In [1]: IDRestart ( )            # Restart the ID (see p5)  – does not change current polarisation

In [1]: Move_ARPES_motor("MotorName", value)    # MotorName= "x", "y", "z", "th"

In [1]: Move_RSXS_motor("MotorName", value)     # MotorName= "x", "y", "z", "th", "tth"
```

For common ARPES/RSXS scripts, see examples (read only):
/home/beams22/29IDUSER/Documents/User_Macros/Examples_ARPES(RSXS).py
If you'd like to look at the details of the BL scripts (read only), see:
/home/beams22/29IDUSER/Documents/User_Macros/ScanFunctions_29id.py

iPython - Scans

In [1]: CA_Average(n) # Turns current amplifier average ON (for n>0) / OFF (n=0)
with n = points to be averaged

In [1]: Scan_Mono_Go(ScanDimension, start, stop, step, *settling_time*†)
Starts mono scan with:
➪ ScanDimension = 1, 2, 3 or 4
➪ settling_time = 0.1s per default (no argument)

In [1]: Scan_Energy_TrackingID_Go(ScanDimension, start, stop, step)
Starts energy scan with tracking ID with:
➪ ScanDimension = 1, 2, 3 or 4
WARNING: very slow !!!! Use Fixed ID if possible:
➪ for energy range > 5% of central photon energy

Aborting a script (ctrl+C) does not abort the current scan record!!!

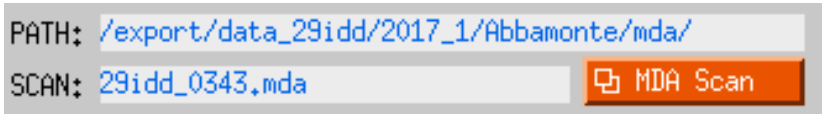
In [1]: Scan_Energy_Go(ScanDimension, start, stop, step, *settling_time*† , *ID_offset*†)
Starts energy scan with fixed ID:
➪ ScanDimension = 1, 2, 3 or 4
➪ settling_time = 1s per default (no argument)
➪ ID energy is set to: (start+stop)/2+ID_offset
➪ ID_offset = 0 per default (no argument)
➪ for energy range < 5% of the photon energy

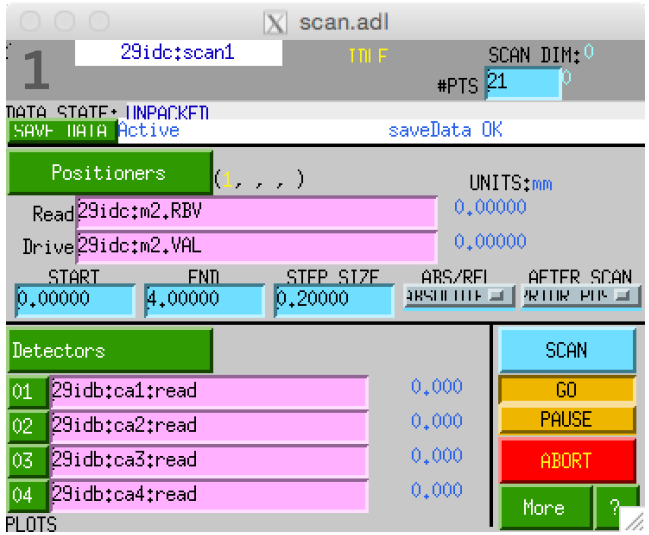
† Optional argument

Notes Scan_Energy_Go:
- RSXS: The scripts automatically drops in the mesh (=[D11] ca14) for normalization I/I_0 ; mesh is automatically retracted at the end of the scan.
- ARPES: At the moment, C branch only has a diode, no mesh; repeat scan with C-diode inserted (=[D12] ca15) to normalize I/I_0
- Settling time is always set back to 0.1s automatically at the end of any scan.

Scan Record

Click on “MDA Scan” button then select “Scan1”:





Scan dimension →

Motor Read Back pv →

Motor Set Point pv →

Start / Stop / Step →

← Number of points
Stays fixed when start/stop changes!

← After scan: stay / prior position / start position…

← START SCAN

← ABORT SCAN

Aborting a script (ctrl+C) does not abort the current scan record

- For 2D scans (e.g. Y/Z map):
  - open Scan2 (outer loop)
  - fill in Scan2 (motor pv read/drive and start/stop/step)
  - start Scan2: Scan2 positioner will move to its first position then trigger Scan1
- For higher dimensionality (e.g. X/Y/X/theta/energy…):
  follow the same steps using Scan3/4/5…
- Start scan by clicking SCAN from highest Scan# (most outer loop)
- Scan1 is <u>always</u> the most inner loop.

dview

File / Load Data
(see mda path
29idd_graphic)

File / Extra PVs:
shows beamline PVs recorded
at the end of each scan

Buffer tab for scan comparisons
(see "bufferize plot" on lower left)

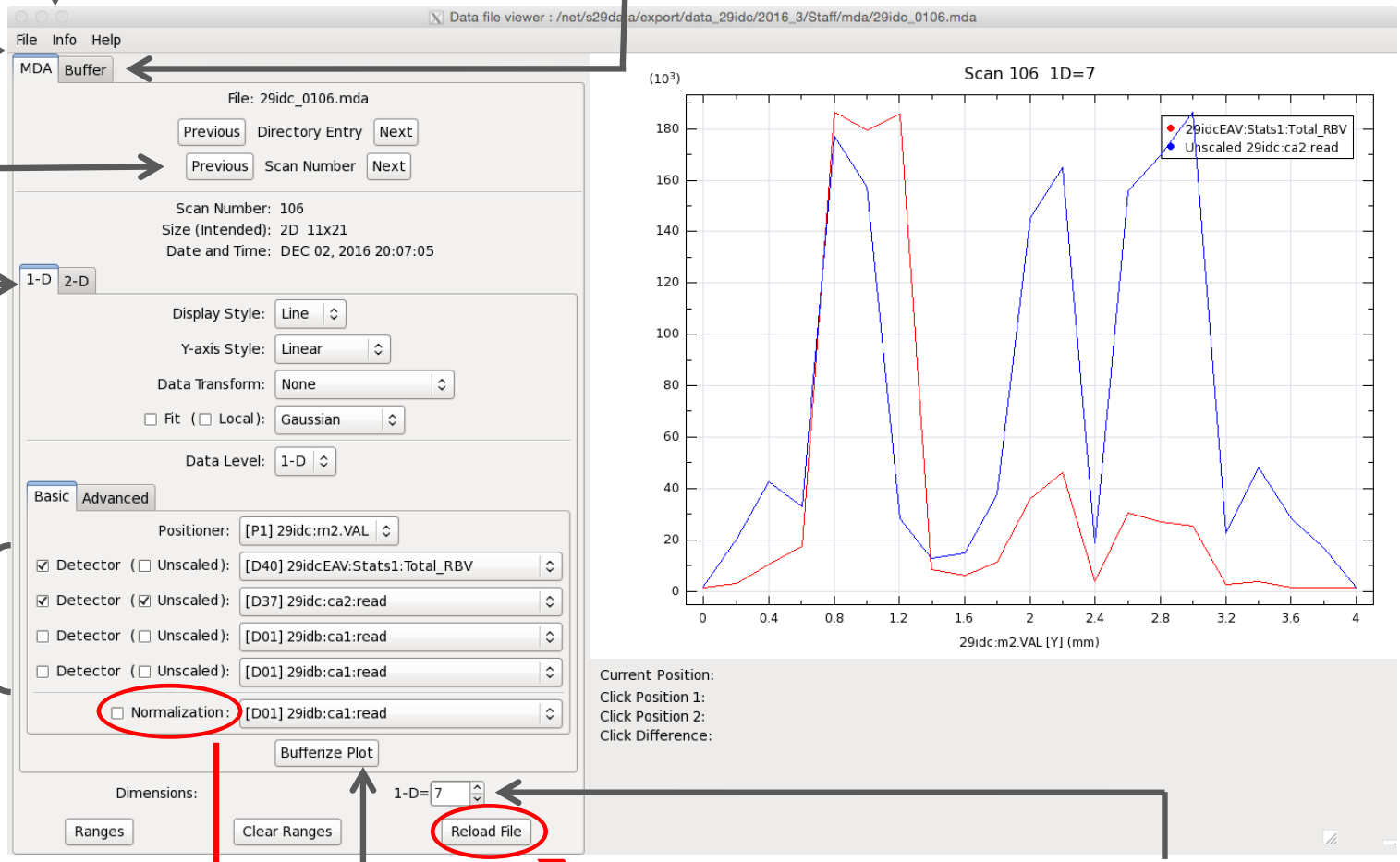
Scroll between scans

1D/2D tabs

Pick detector to plot

TEY [D13]
EA [D14]
MPA [D15]
TEY [D16]
D-1 [D17]
D-2 [D18]

Diode [D12] (I_0)
Mesh [D11] (I_0)



Bufferize plot to compare ≠ scans

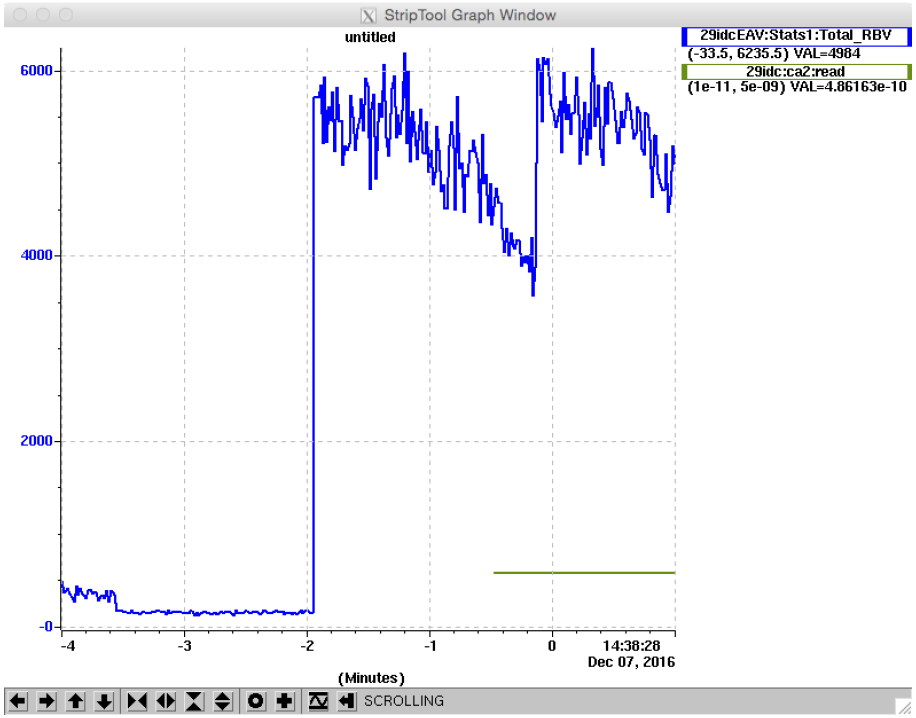
During scan acquisition, "Reload file" to update

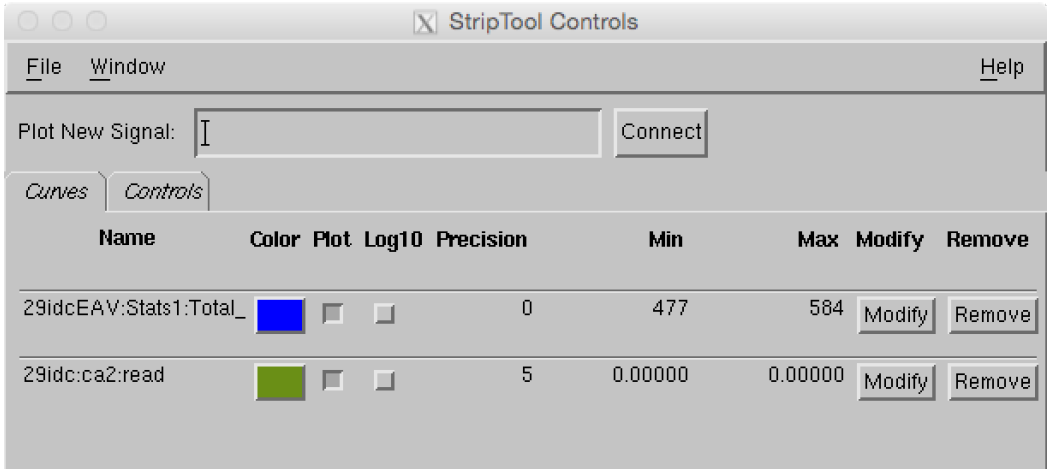
2D scans only: change higher dimensional values (in point index); to display actual positioner value: info/higher dimensional values

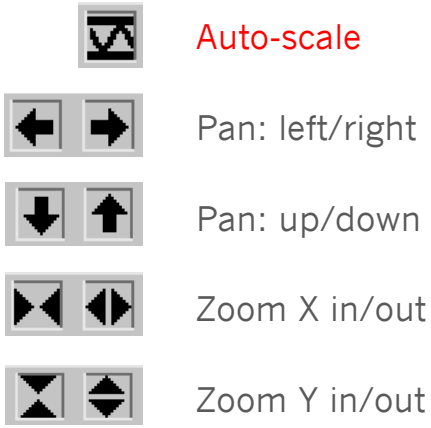
Striptool

Striptool allows to monitor a PV over time. It is useful to optimize sample position while looking at the count rate:

- In ! Tools select Striptool
- "Plot New Signal": enter PV to monitor, e.g.:
  - TEY (ARPES): *29idc:ca2:read*
  - EAV (ARPES): 29idcEAV:Stats1:Total_RBV
  - TEY (RSXS): *29idd:ca2:read*
  - Small diode (RSXS): *29idb:ca16:read*
- Click "Connect"
- Can monitor several PVs at the same time.







Beam Dump

- When the beam dumps:
  - Status button turns red: ID permission is switched to “Operator”
  - Main shutter is disable
  - Check Infos for updates / time estimate
  - iPython scripts will pause when trying to change energy and/or start a new scan – i.e. you will only loose the one scan that was running at the time of the beam dump

- After a beam dump (control room’s loud speaker announcement):
  - Check main shutter is open
  - Check ID status button (p5):
    - if red, the ID is not in user mode: call floor coordinator (2-0101, then dial 2601 when prompt) or control room (2-9424)
    - if black, the ID is off: use Python script *StartID(mode)* with mode = “H”, “V”, ”RCP”, ”LCP”
  - Check ID polarization & energy are correct !!!

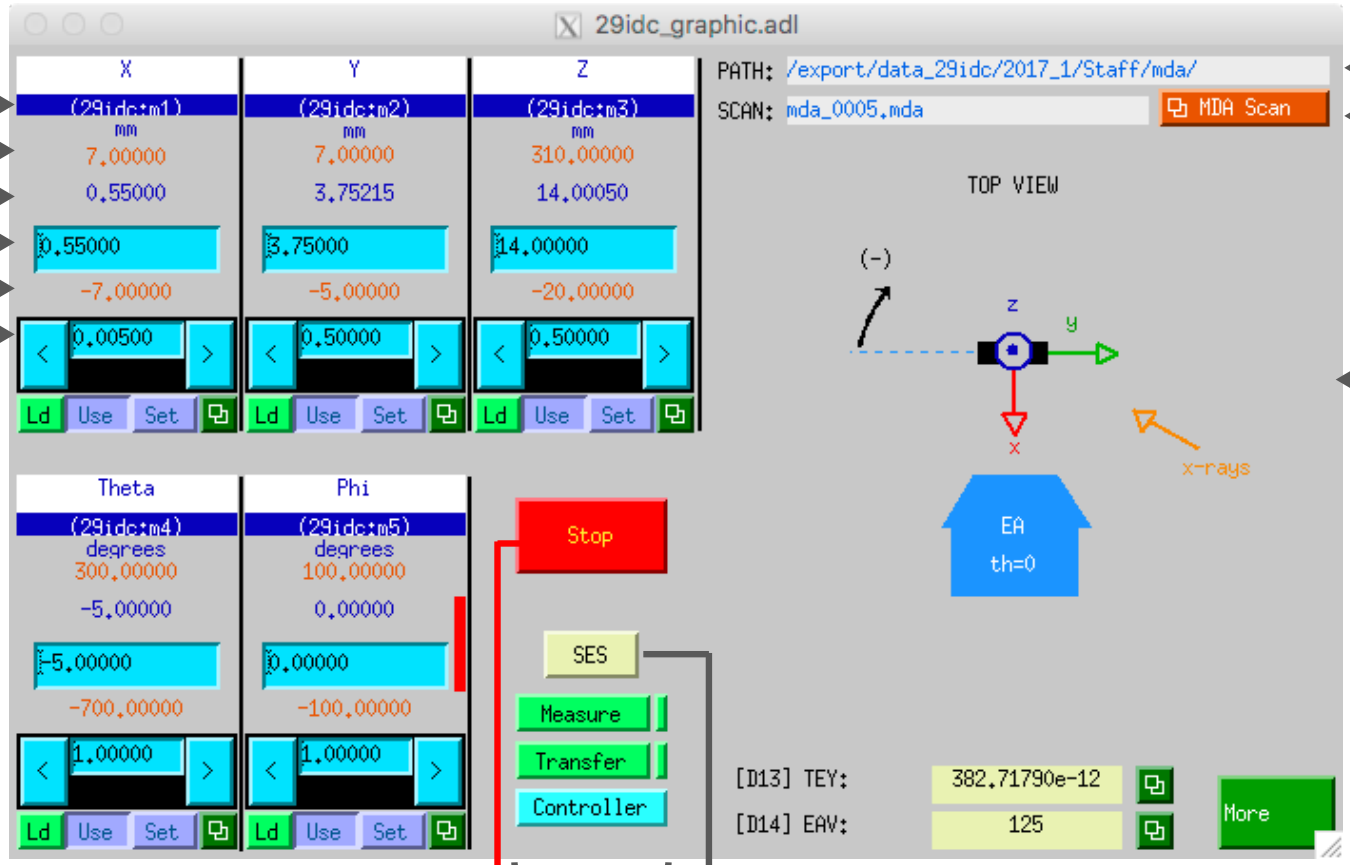
Troubleshooting: No counts?

- Checklist:
  - Is there current in the ring?
  - Are all the shutters/valves opened? ➪ beamline sketch should be green all the way up to the end-station
  - Is there a diagnostic blocking the beam? ➪ see ”Diagnostics” button - click “out” button on BL screen
  - Drop mesh/diode in, does it read current? (don’t forget to pull it out!)
  - Did you try to open-up the exit slits?
  - Do the Mono & ID readback values match each other (within ~10%)?
  - Is there TEY current?
  - Is the sample reasonably close to measuring position? (check camera)
  - MPA / SES : Is the HV ON?
  - ARPES : Is the EA viewer ON? Is the Kinetic Energy appropriate?
  - Check slits 1A/2B: click 

, all slits should be of the order of millimeters and centered at 0

- If all of the above is true:
  - Tweak mono energy to optimize current on TEY
  - Refine sample position by performing a 2 or 3D scan
  - ARPES : Increase Pass Energy and/or analyzer slit to increase count rate; go to a deep core level

ARPES manipulator



motor pv
upper limit
readback
set point
lower limit
tweak
buttons

Data folder & scan#

Open scan record (p9)

X // nose cone
Y ⊥ nose cone
For all θ angles

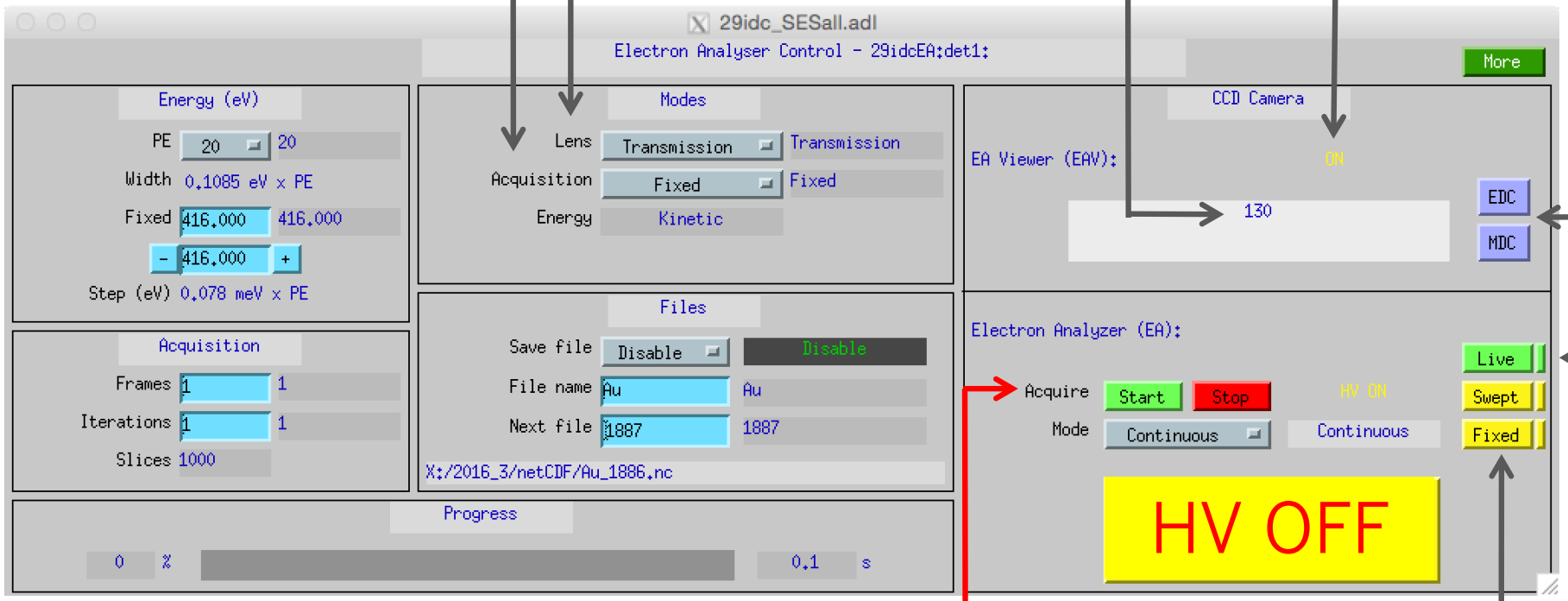
Emergency stop:
stops ALL ARPES motors

preset positions

Scienta screen

TEY (29idc:ca2:read) : drain current
metallic sample only !
EAV (29idcEAV:Stats1:Total_RBV) : total
count rate (arb. unit)

Electron Analyzer



Acquisition mode: fixed or swept

Lens mode: transmission or angular

Count rate: max 300,000 (HV is interlocked)

Live viewer ON/OFF

Open MDC/EDC screens
To auto-scale, right click anywhere in the window: Y1 Axis / Axis Range / Auto-scale

Scan progress

Time estimate for current setting: frames, iterations, energy step, mode (swept/fixed)

Scan: Start / Stop
HV: On / Off

Switch to scan mode - fixed or swept
i.e. saving ON & single scan mode

Switch to live mode
i.e. saving OFF & continuous scan mode w/ PE 20

Scan Settings

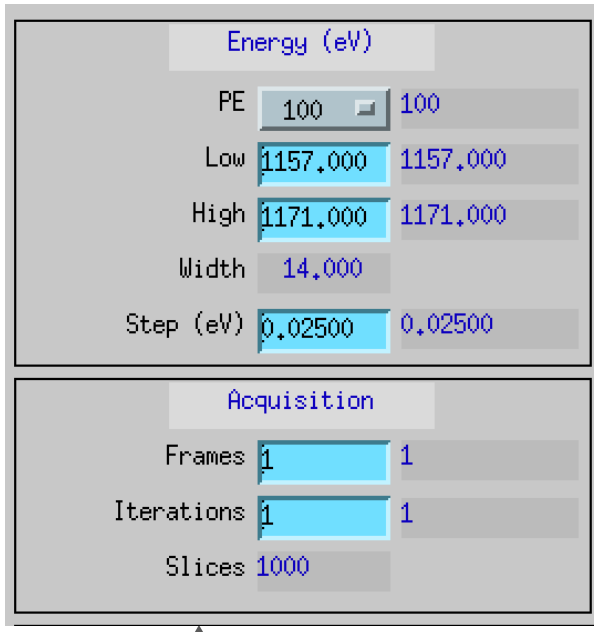
Kinetic Energy only:
$E_K = h\nu - E_B$

Work function ϕ is already implemented
i.e. E_F @ $E_B = 0$

Turn off HV before changing PE

Swept Mode:

E_K range (min) →
E_K range (up) →
E_k window width →
E_K Steps →

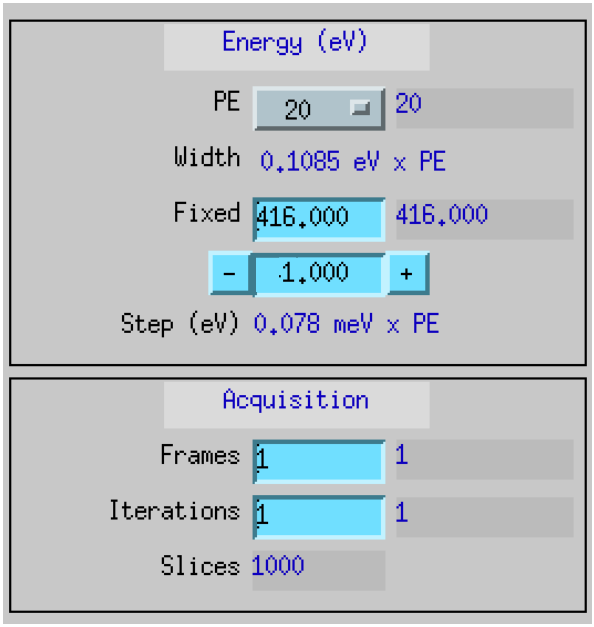


At the moment slices # is fixed to 1000
(1 pixel = 1 slice)
Angular mode acceptance = 14°
Different Angular modes have different energy ranges vs PE. Refer to the following table:

Fixed Mode:

Pass Energy →
E_K window width →
E_K window center →
Tweak center →
E_K Steps →

Exposure time
(1 f = 1/17 s) →
Iterations # →



XPS Mode	1	2	5	10	20	50	100	200	500
Trans	0-76	0-107	1-237	1-453	3-968	7-6041	21-6000	30-6206	283-6504
Angular		1-52	2-131	5-262	10-523	24-1309	34-6105	230-6206	
Angular_1						126-2947	252-1401	844-1261	
Angular_5							252-1323	1043-1270	

Be very careful when using relative scans

iPython - ARPES

In [1]: Scan_ARPES_motor_Go("MotorName", start, stop, step, *settling_time†, mode†*)

Starts motor scan:

- MotorName= "x", "y", "z", "th"
- settling_time = 0.1s per default (no argument)
- mode = "relative" for relative scan; "absolute" per default (no arg.)

In [1]: Scan_ARPES_2Dmotor_Go("MotorName1", start1, stop1, step1, "MotorName2", start2, stop2, ... step2, *settling_time†, mode†*)

Starts 2D motor scan:

- MotorName= "x", "y", "z", "th"
- settling_time = 0.1s per default (no argument)
- mode = "relative" for relative scan; "absolute" per default (no arg.)

In [1]: Move_ARPES_Sample(ListMotor)

Move to a given position define by ListMotor:

- ListMotor = ["Sample Name" , x, y, z, th, *coeff †*]
- coeff = adjust number of iteration for a given position (see examples)
  value set to 1 per default

In [1]: Resolution_ARPES(grating, energy, Exit Slit, PE , slit_SES , T(K))

Calculate the theoretical resolution

- grating = "HEG" or "MEG"
- Exit Slit = 10, 20, 50, 100, 200
- PE = Pass Energy
- Slit_SES = 1 to 9 (see slit table in control room)

† Optional argument

iPython - ARPES

```
In [1]: EASwept("LowHigh", "BE", E1, E2, step, PE, frames, i, "transmission")
        # Start Scan in Swept mode with the following arguments:
                [0] Range_mode: "Center" or "LowHigh"          [5] Pass Energy
                [1] Energy_mode: "KE" or "BE" - BE = hv-KE     [6] Frames
                [2] Center or E1                               [7] Iterations
                [3] Width  or E2                               [8] "transmission" / "angular"
                [4] Step size

In [1]: EASwept("BE", Center, PE, frames, i, "transmission")
        # Start Scan in Fixed mode with the following arguments:
                [0] Energy_mode: "KE" or "BE" - BE = hv-KE     [3] Frames
                [1] Center                                     [4] Iterations
                [2] Pass Energy                                [5] "transmission" / "angular"

In [1]: FermiMap(ListPositions,ListEA)
                # Start a series of scan for a given number of sample positions defined in ListPositions

In [1]: KzMap(start,stop,step,ListEA)
                # Start a series of scan vs photon energy
```

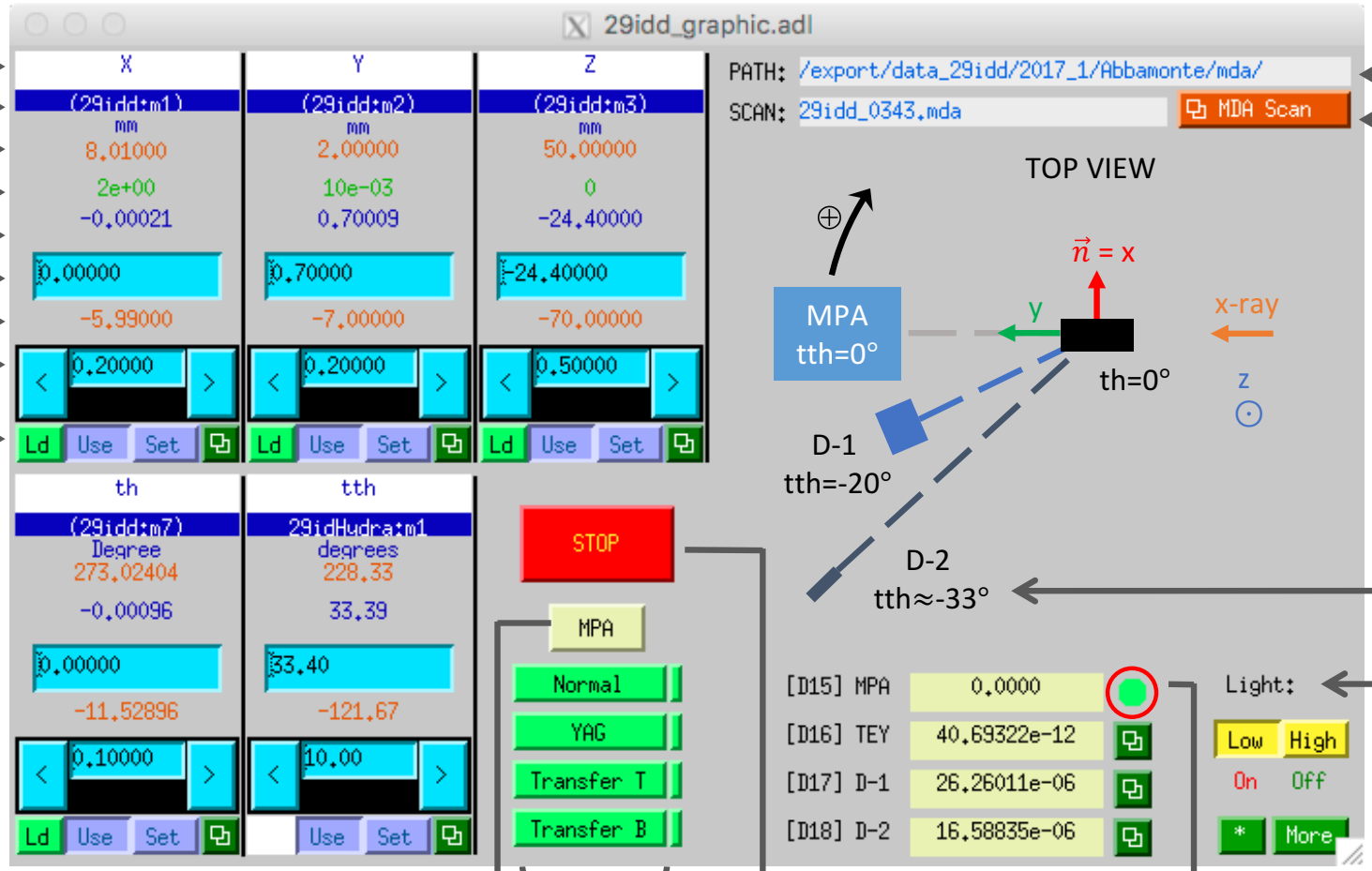
For common ARPES scripts, see examples (read only):
/home/beams22/29IDUSER/Documents/User_Macros/Examples_ARPES.py

† Optional argument

Electron Analyzer crash

- The epics version of the Electron Analyzer data acquisition is running from a Windows machine named “Glick”.
- If the EA screens fields turn white, it is because it crashed. To restart it, open a remote session to glick:
  - Open a terminal
  - Type: *glick_29id*
  - Enter password: *ask staff*
  - On Glick, close the (black) terminal window - make sure it closes properly!
  - Double click on the shortcut “start_ioc.bat” on the lower left corner of the desktop.
  - A new terminal window pops-up, the EA screen should be ready to use within a few seconds.

RSXS manipulator



name
motor pv
upper limit
encoder
readback
set point
lower limit
tweak buttons
load scan button

Data folder & scan#

Open scan record (p9)

Diodes **position** when MPA is in direct beam (th=0°)

Chamber light
Turn OFF to use diode or MCP

Area detector screen

preset positions

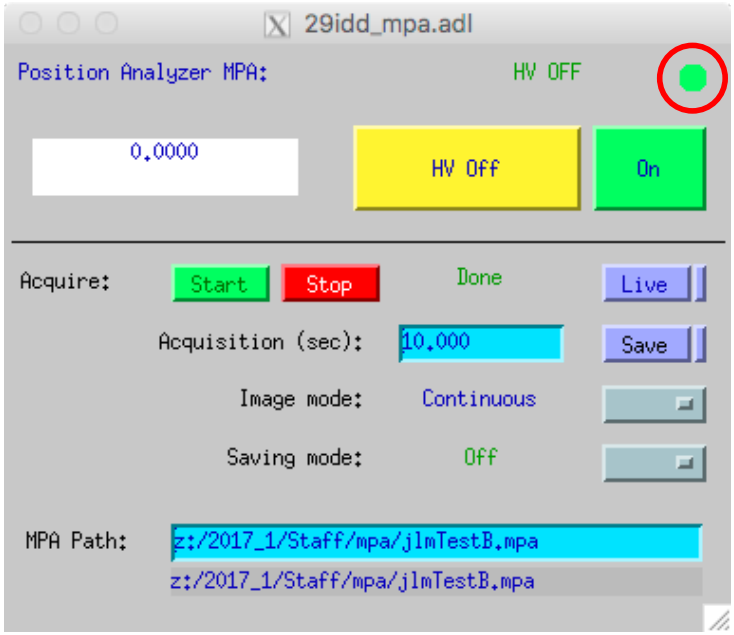
Emergency stop:
stops ALL RSXS motors

MPA Status
(see next page)

mini YAG
permanently mounted below sample holder
X is very different from sample – can be confusing!

MPA (29iddau1:dau1:005:ADC) : total count rate (arb. unit)
TEY (29idd:ca2:read) : drain current (metallic sample only !)
D-1 (29idd:ca3:read) : large diode (ang. acceptance ~ 3°)
D-2 (29idb:ca16:read) : small diode (ang. acceptance ~ 0.14°)
Mesh (29idb:ca14:read) : for I_0 normalization

MPA Area Detector



Count rate:
max 7.3
(HV is interlocked) →

Scan: Start / Stop →

Acquisition Time
(single image mode) →

File Path →

← MPA status (see below)

← HV ON/OFF

Acquisition mode:
- Live: saving off / continuous
- Save: saving on / single

Switch buttons:
- mode: single / continuous
- saving: on / off

MPA Status:

- OK to turn on HV if P < $1e^{-7}$ Torr
- Detector in direct beam **or** count went above authorized limit:
  - ➪ 7.3 on screen (i.e. ~10^5 Hz)
  - ➪ HV off to protect the MPA
- Not OK - call staff

iPython - RSXS

Be very careful when using relative scans

In [1]: Move_RSXS_Sample(ListMotor)
Move to a given position define by ListMotor:
➪ ListMotor = ["Sample Name" , x, y, z, th, *tth†*]

In [1]: Move_th2th(th_value, tth_offset)
where tth_offset is the offset between given detector and direct beam:
➪ MPA: offset = 0
➪ D1-D: offset = 20
➪ D2-D: offset = 33ish – to be refined regularly

In [1]: Scan_RSXS_motor_Go("MotorName", start, stop, step, *settling_time†, mode†*)
Starts motor scan:
➪ MotorName= "x", "y", "z", "th", "tth"
➪ settling_time = 0.1s per default (no argument)
➪ mode = "relative" for relative scan; "absolute" per default (no arg.)

In [1]: Scan_RSXS_2Dmotor_Go("MotorName1", start1, stop1, step1, "MotorName2", start2, stop2,
... step2, *settling_time†, mode†*)
Starts 2D motor scan:
➪ MotorName= "x", "y", "z", "th", "tth"
➪ settling_time = 0.1s per default (no argument)
➪ mode = "relative" for relative scan; "absolute" per default (no arg.)

In [1]: Scan_th2th_Go(start, stop, step, tth_offset, *settling_time†*)
Starts th-tth scan for a given tth_offset:
➪ MPA: offset = 0
➪ D1-D: offset = 20
➪ D2-D: offset = 33ish – to be refined regularly
assuming sample is aligned at th=0 in grazing incidence

† Optional argument

Align Sample - RSXS

- In normal incidence (preset position Normal):
  - Set Y /Z using cursor on camera
  - Refine by doing (Y,Z) 1D scan and/or 2D map for inhomogeneous samples

- In grazing incidence (th=0):
  - Scan X to find center of the beam X_0:
    ➪ reset X to 0 at ½ intensity
  - Scan th (coarse rocking curve: -4/4/0.1) to find th_0 using D-2:
    ➪ reset th = 0 at maximum intensity
  - Repeat until X_0 and th_0 converge

- Go to th-2th condition (e.g. th=5, tth=10+$Diode2_{offset}$):
  ➪ *Move_th2th(5, $Diode2_{offset}$)*
  ➪ *Scan_RSXS_motor_Go("th",-0.5,0.5,0.01,"relative")*
  ➪ determine center position using gaussian fit
  ➪ set center position to (tth-$Diode2_{offset}$)/2
  ➪ refine X_0 if needed.