psana - Module Examples

Unknown macro: 'html'

Unknown macro: 'html'

About

This page provides examples for selected modules from psana - Module Catalog.

Auxiliary scripts

A few python scripts in `ImgPixSpectra/data/` show how to process/plot the spectral array stored in the file.

- `PlotSpectralArrayFromFile.py` - allows to plot content of the spectral array as a 2-d plot.

- `SpectralArray.py` - provides access to the spectral array stored in the file. The `class SpectralArray` defined in this script is used in the `PlotSpectralArrayFromFile.py`.
- `MergeArrays.py` - sums the arrays from different files defined in the list and saves resulting array in a single file with the same shape. In this script the list of files is hardwired in the `get_list_of_input_file_names()` method. The output file name, `out_fname`, is also hardwired in the call to `spectra_merging(out_fname)`.

A few auxiliary scripts for example are located in the directory `ImgAlgos/data`:

- `psana.cfg` - set non-default parameters to run `psana` for `ImgAlgos::ImgPeakFinder` and `ImgAlgos::ImgPeakFilter`
  The `psana` running this script saves images and peaks for event 115 in text files.
- `PlotCameraImageFromFile.py` - Plots image and spectrum for image saved in file.
- `PlotCameraImageAndPeaks.py` - Plots image with found peaks and spectrum.

Examples for package cspad_mod

Example for module cspad_mod::CsPad2x2Pedestals

Configuration file for pedestals calibration of two CSPAD2x2 simultaneously:

```
[psana]
files = /reg/d/psdm/xpp/xpptut13/xtc/e308-r0070-s02-c00.xtc \
        /reg/d/psdm/xpp/xpptut13/xtc/e308-r0070-s03-c00.xtc

modules = cspad_mod.CsPad2x2Pedestals:0 \
          cspad_mod.CsPad2x2Pedestals:1

#skip-events = 100
#events = 1111

[cspad_mod.CsPad2x2Pedestals:0]
source = DetInfo(XppGon.0:Cspad2x2.0)
output = pedestals-ave-xppi0513-r0070-Cspad2x2.0.dat
noise  = pedestals-rms-xppi0513-r0070-Cspad2x2.0.dat

[cspad_mod.CsPad2x2Pedestals:1]
source = DetInfo(XppGon.0:Cspad2x2.1)
output = pedestals-ave-xppi0513-r0070-Cspad2x2.1.dat
noise  = pedestals-rms-xppi0513-r0070-Cspad2x2.1.dat
```

Command to run this script:

```
psana -c psana-xppi0513-r0070-cspad2x2-pedestals.cfg
```

The xtc file name(s) may be passed as an argument:

```
psana -c psana-xppi0513-r0070-cspad2x2-pedestals.cfg /reg/d/psdm/xpp/xpptut13/xtc/e308-r0070-*.xtc
```

Output files contain results, which can be plotted for average values:

and rms values:

Examples for package CSPadPixCoords

Example for module CSPadPixCoords::CSPadImageProducer

How to write the CSPad image in text file:

```
[psana]
files         = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-1>.xtc
events        = 5

modules       = cspad_mod.CsPadCalib CSPadPixCoords.CSPadImageProducer ImgAlgos.ImgSaveInFile

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = no
doCommonMode  = no

#[CSPadPixCoords.CSPadInterpolImageProducer]
[CSPadPixCoords.CSPadImageProducer]
calibDir      = /reg/d/psdm/<instrument>/<experiment>/calib
typeGroupName = CsPad::CalibV1
source        = CxiDs1.0:Cspad.0
key           = calibrated
imgkey        = Image2D
tiltIsApplied = true

[ImgAlgos.ImgSaveInFile]
source        = CxiDs1.0:Cspad.0
#eventSave    = 1
saveAll       = true
```

See Package CSPadPixCoords

Example for module CSPadPixCoords::CSPadNDArrProducer

See Module CSPadPixCoords::CSPadNDArrProducer

Example of the module CSPadNDArrProducer working in sequence with ImgAlgos::NDArrAverage is shown in Example for module ImgAlgos::NDArrAverage

Example for module CSPadPixCoords::CSPad2x2NDArrProducer

```
# Run this script:
# psana -c psana-meca1113-r0045-cspad-cspad2x2-dark-hotpix.cfg

[psana]
# Default calibration directory:
# calib-dir = /reg/d/psdm/mec/meca1113/calib

files = exp=meca1113:run=45
events = 400
#skip-events = 0

modules = CSPadPixCoords.CSPad2x2NDArrProducer:1 \
          CSPadPixCoords.CSPad2x2NDArrProducer:2 \
          ImgAlgos.NDArrAverage:1 \
          ImgAlgos.NDArrAverage:2


[CSPadPixCoords.CSPad2x2NDArrProducer:1]
source  = MecTargetChamber.0:Cspad2x2.1
inkey   =
outkey  = cspad2x2.1_ndarr
outtype = int16
print_bits  = 3


[CSPadPixCoords.CSPad2x2NDArrProducer:2]
source  = MecTargetChamber.0:Cspad2x2.2
inkey   =
outkey  = cspad2x2.2_ndarr
outtype = int16
print_bits  = 3


[ImgAlgos.NDArrAverage:1]
source       = MecTargetChamber.0:Cspad2x2.1
key          = cspad2x2.1_ndarr
avefile      = cspad2x2.1-ave
rmsfile      = cspad2x2.1-rms
maskfile     = cspad2x2.1-msk
hotpixfile   = cspad2x2.1-hot
#evts_stage1  = 100
#gate_width1  = 100.
thr_rms_ADU  = 10
thr_min_ADU  = 4
thr_max_ADU  = 10000
print_bits   = 29


[ImgAlgos.NDArrAverage:2]
source       = MecTargetChamber.0:Cspad2x2.2
key          = cspad2x2.2_ndarr
avefile      = cspad2x2.2-ave
rmsfile      = cspad2x2.2-rms
maskfile     = cspad2x2.2-msk
hotpixfile   = cspad2x2.2-hot
#evts_stage1  = 100
#gate_width1  = 100.
thr_rms_ADU  = 10
thr_min_ADU  = 4
thr_max_ADU  = 10000
print_bits   = 29
```

This script makes ndarrays for two cspad2x2 detectors and use them evaluate average, rms, mask and hot pixel arrays.

See Module CSPadPixCoords::CSPad2x2NDArrProducer

Example for module CSPadPixCoords::CSPad2x2NDArrReshape

Example of configuration file

```
[psana]
#files = exp=meca1113:run=376
#events = 10
##skip-events = 0

modules = cspad_mod.CsPadCalib \
          CSPadPixCoords.CSPad2x2NDArrProducer:clb \
          CSPadPixCoords.CSPad2x2NDArrProducer:raw \
          CSPadPixCoords.CSPad2x2NDArrReshape \
          ImgAlgos.NDArrAverage:clb \
          ImgAlgos.NDArrAverage:raw
#          EventKeys

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = clb_data
doPedestals   = yes
doPixelStatus = yes
doCommonMode  = yes

[CSPadPixCoords.CSPad2x2NDArrProducer:clb]
source     = MecTargetChamber.0:Cspad2x2.1
inkey      = clb_data
outkey     = cspad2x2.1_clb:as_data
outtype    = int16
print_bits = 5

[CSPadPixCoords.CSPad2x2NDArrProducer:raw]
source     = MecTargetChamber.0:Cspad2x2.1
inkey      =
outkey     = cspad2x2.1_raw:as_data
outtype    = int16
print_bits = 5

[CSPadPixCoords.CSPad2x2NDArrReshape]
source     = MecTargetChamber.0:Cspad2x2.1
keys_in    = cspad2x2.1_raw:as_data cspad2x2.1_clb:as_data
print_bits = 255

[ImgAlgos.NDArrAverage:clb]
source       = MecTargetChamber.0:Cspad2x2.1
key          = cspad2x2.1_clb
avefile      = arr-ave-clb
rmsfile      = arr-rms-clb
print_bits   = 255

[ImgAlgos.NDArrAverage:raw]
source       = MecTargetChamber.0:Cspad2x2.1
key          = cspad2x2.1_raw
avefile      = arr-ave-raw
rmsfile      = arr-rms-raw
print_bits   = 255
```

This script produces raw and calibrated ndarrays shaped as data (185,388,2), then pass these arrays as

> *keys_in   = cspad2x2.1_raw:as_data cspad2x2.1_clb:as_data*

to the CSPadPixCoords.CSPad2x2NDArrReshape module, which produces two reshaped (2,185,388) arrays and saves them in the event store with default keys

> *cspad2x2.1_raw, cspad2x2.1_clb (suffixes are dropped)*

then both re-shaped arrays are averaged and saved in files on disk.

Example of associated python script:

```
#!/usr/bin/env python

import sys
import numpy as np
import matplotlib.pyplot as plt
import psana

psana.setConfigFile('psana-meca1113-r0376-cspad2x2-CSPad2x2NDArrReshape.cfg')

dsname = 'exp=meca1113:run=376'
print "Data source: %s" % dsname
ds = psana.DataSource(dsname)

#------------------------------
print "Initializing Matplotlib Plotter"
fig = plt.figure(figsize=(10,5), dpi=80, facecolor='w', edgecolor='w', frameon=True)
plt.ion()
plt.show()

evnum = 0
evnum_max = 50

for evt in ds.events() :

    evtid  = evt.get(psana.EventId)

    evnum += 1
    if evnum > evnum_max : break

    img_as_data = evt.get(psana.ndarray_int16_3, psana.Source('DetInfo(MecTargetChamber.0:Cspad2x2.1)'),
'cspad2x2.1_clb:as_data')
    img_reshpd  = evt.get(psana.ndarray_int16_3, psana.Source('DetInfo(MecTargetChamber.0:Cspad2x2.1)'),
'cspad2x2.1_clb')

    print 'img_as_data.shape = ', img_as_data.shape
    print 'img_reshpd.shape  = ', img_reshpd.shape

    img_as_data.shape = (2*185,388)
    img_reshpd.shape = (2*185,388)

    ax1 = fig.add_axes([0.05,  0.06, 0.44, 0.87])
    ax2 = fig.add_axes([0.55,  0.06, 0.44, 0.87])

    fig.canvas.set_window_title('Image from arrays "as-data" and "reshaped"')

    imsh1 = ax1.imshow(img_as_data, interpolation='nearest', aspect='auto', origin='upper') # ,
extent=img_range)
    imsh2 = ax2.imshow(img_reshpd,  interpolation='nearest', aspect='auto', origin='upper') # ,
extent=img_range)

    plt.title("Event: %d.%d" % evtid.time())
    plt.draw()
    plt.clf()
```

This script retrieves from the event store calibrated ndarrays shaped as data (185,388,2) and re-shaped (2,185,388), then re-shape both to 2-d arrays (2*185,388) and plot them. Re-shaped array looks as recognizable (without spaces between ASICs) cspad2x2 image. Array shapes "as data" looks as an overlap of two images.

See Module CSPadPixCoords::CSPad2x2NDArrReshape

JIRA issue: **PSAS-45** - Getting issue details... STATUS

Example for module CSPadPixCoords::CSPad2x2ImageProducer

See Module CSPadPixCoords::CSPad2x2ImageProducer

Example of the configuration script for psana (cspad2x2-test.cfg):

```
[psana]
files         = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-1>.xtc
events        = 5
modules       = CSPadPixCoords.CSPad2x2ImageProducer ImgAlgos.ImgSaveInFile

[CSPadPixCoords.CSPad2x2ImageProducer]
source         = DetInfo(:Cspad2x2)
inkey          =
outimgkey      = Image
tiltIsApplied  = true
print_bits     = 15

[ImgAlgos.ImgSaveInFile]
source        = DetInfo(:Cspad2x2)
key           = Image
fname         = cspad2x2
saveAll       = true
#eventSave     = 5
```

Command to run:
`psana -c cspad2x2-test.cfg`
One of the saved files `cspad2x2-<run>-<timestamp>.txt` is plotted as an image by the command
`./PlotCameraImageFromFile.py cspad2x2-<run>-<timestamp>.txt 0 1200`

Configuration file for cspad2x2 with pedestal subtraction

> ⚠ Access to the CSPad2x2 aligned geometry is added on 2013-02-13
> and is available for offline release > `ana-0.9.1`.

Example of <psana-config-file.cfg>:

```
[psana]
files    = /reg/d/psdm/mec/mec73313/xtc/e268-r0180-s02-c00.xtc
#calib-dir = ./calib

modules  = cspad_mod.CsPadCalib CSPadPixCoords.CSPad2x2ImageProducer ImgAlgos.ImgSaveInFile
events   = 5

[cspad_mod.CsPadCalib]
source        = DetInfo(MecTargetChamber.0:Cspad2x2.3)
inputKey      =
outputKey     = calibrated_arr
doPedestals   = yes
doPixelStatus = no
doCommonMode  = no


[CSPadPixCoords.CSPad2x2ImageProducer]
calibDir       = /reg/d/psdm/mec/mec73313/calib
typeGroupName  = CsPad2x2::CalibV1
#source        = DetInfo(MecTargetChamber.0:Cspad2x2.3)
source         = :Cspad2x2.3
inkey          = calibrated_arr
outimgkey      = Image
tiltIsApplied  = true
print_bits     = 15


[ImgAlgos.ImgSaveInFile]
source         = DetInfo(MecTargetChamber.0:Cspad2x2.3)
key            = Image
fname          = cspad2x2.3
saveAll        = true
print_bits     = 3
#eventSave     = 5
```

Example of psana configuration file to get cspad2x2 images for two detectors and save them in files, one in txt, another in tiff formats:

```
[psana]
files    = /reg/d/psdm/xpp/xpptut13/xtc/e308-r0008-s02-c00.xtc \
           /reg/d/psdm/xpp/xpptut13/xtc/e308-r0008-s03-c00.xtc

#modules = cspad_mod.CsPad2x2Pedestals

#calib-dir = ./calib
calib-dir = /reg/d/psdm/xpp/xpptut13/xtc/calib

modules  = cspad_mod.CsPadCalib:0 \
           cspad_mod.CsPadCalib:1 \
           CSPadPixCoords.CSPad2x2ImageProducer:0 \
           CSPadPixCoords.CSPad2x2ImageProducer:1 \
           ImgAlgos.ImgSaveInFile:0  \
           ImgAlgos.ImgSaveInFile:1

events   = 5

[cspad_mod.CsPadCalib:0]
source        = DetInfo(XppGon.0:Cspad2x2.0)
inputKey      =
outputKey     = calibrated_arr0
doPedestals   = yes
doPixelStatus = no
doCommonMode  = yes

[cspad_mod.CsPadCalib:1]
source        = DetInfo(XppGon.0:Cspad2x2.1)
inputKey      =
outputKey     = calibrated_arr1
doPedestals   = yes
```

```
doPixelStatus = no
doCommonMode  = yes


[CSPadPixCoords.CSPad2x2ImageProducer:0]
calibDir         = /reg/d/psdm/xpp/xpptut13/xtc/calib
typeGroupName    = CsPad2x2::CalibV1
source           = DetInfo(XppGon.0:Cspad2x2.0)
inkey            = calibrated_arr0
outimgkey        = Image
tiltIsApplied    = false
useWidePixCenter = false
print_bits       = 15

[CSPadPixCoords.CSPad2x2ImageProducer:1]
calibDir         = /reg/d/psdm/xpp/xpptut13/xtc/calib
typeGroupName    = CsPad2x2::CalibV1
source           = DetInfo(XppGon.0:Cspad2x2.1)
inkey            = calibrated_arr1
outimgkey        = Image
tiltIsApplied    = false
useWidePixCenter = false
print_bits       = 15



[ImgAlgos.ImgSaveInFile:0]
source         = DetInfo(:Cspad2x2.0)
key            = Image
fname          = cspad2x2.0
ftype          = txt
#ftype          = tiff
saveAll        = true
print_bits     = 3
#eventSave     = 5

[ImgAlgos.ImgSaveInFile:1]
source         = DetInfo(:Cspad2x2.1)
key            = Image
fname          = cspad2x2.1
#ftype          = txt
ftype          = tiff
saveAll        = true
print_bits     = 3
#eventSave     = 5
```

Example for package ImgPixSpectra

See Package ImgPixSpectra

Modules:

- ImgPixSpectra::CSPadPixSpectra
- ImgPixSpectra::CSPad2x2PixSpectra
- ImgPixSpectra::CameraPixSpectra

Example for module ImgPixSpectra::CSPadPixSpectra

See module description in Module ImgPixSpectra::CSPadPixSpectra
Configuration file `psana-cxib2313-r0114-cspad-pix-spectra.cfg`:

```
# Command to run this script:
# psana -c psana-cxib2313-r0114-cspad-pix-spectra.cfg
#
# Other useful commands:
# psana -n 5 -m  PrintSeparator,PrintEventId,psana_examples.DumpCsPad /reg/d/psdm/cxi/cxib2313/xtc/e336-r0114*
# psana -n 5 -m EventKeys exp=cxib2313:run=114:xtc

[psana]
files      = exp=cxib2313:run=114:xtc
#calib-dir = ./calib
skip-events = 0
events      = 100
modules     = cspad_mod.CsPadCalib ImgPixSpectra.CSPadPixSpectra

[cspad_mod.CsPadCalib]
source        = DetInfo(CxiDs1.0:Cspad.0)
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = no
doCommonMode  = yes

[ImgPixSpectra.CSPadPixSpectra]
source        = CxiDs1.0:Cspad.0
inputKey      = calibrated
amin          =   -20.
amax          =    20.
nbins         =    10
arr_fname     = cspad_spectral_array.txt
```

where module `cspad_mod.CsPadCalib` subtracts pedestals, apply common mode correction, and save CSPAD array in the event store with key "calibrated", which is used in the next module `ImgPixSpectra.CSPadPixSpectra`.

To run this script use command

> *psana -c psana-cxib2313-r0114-cspad-pix-spectra.cfg*

which produces two files:

- `cspad_spectral_array.txt` – array of 10-bin amplitude spectra for all pixels
- `cspad_spectral_array.txt.sha`– file with metadata for array shape:

```
NPIXELS  2296960
NBINS    10
AMIN     -20
AMAX     20
NEVENTS  100
ARRFNAME cspad_spectral_array.txt
```

Plot for content of the file `cspad_spectral_array.txt`:

Example for module ImgPixSpectra::CSPad2x2PixSpectra

See module description in Module ImgPixSpectra::CSPad2x2PixSpectra
Configuration file example for `CSPad2x2PixSpectra`:

```
[psana]
files         = /reg/d/psdm/<instrument>/<experiment>/<file-name>.xtc
modules       = ImgPixSpectra.CSPad2x2PixSpectra

[ImgPixSpectra.CSPad2x2PixSpectra]
source        = CxiSc1.0:Cspad2x2.0
amin          =   500.
amax          =  1000.
nbins         =   100
arr_fname     = cspad2x2-pix-spectra.txt
```

To get images from saved file one may execute the auxiliary script:

```
ImgPixSpectra/data/PlotSpectralArrayFromFile.py cspad2x2-pix-spectra.txt
```

generates image for limited range of pixels for CSPad, CSPad2x2, or Camera, respectively:

Examples for package ImgAlgos

See Package ImgAlgos

Example for module ImgAlgos::Tahometer

See Module ImgAlgos::Tahometer

Example of the psana configuration file:

```
[psana]
files         = /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/<file-name-1>.xtc\
                /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/<file-name-2>.xtc

modules       = ImgAlgos.Tahometer

[ImgAlgos.Tahometer]
dn         = 10
print_bits = 7
```

Example for module ImgAlgos::EpixNDArrProducer

See Module ImgAlgos::EpixNDArrProducer

Example of the psana configuration file:

```
[psana]
files         = exp=xppi0414:run=94
events        = 100
modules       = ImgAlgos.EpixNDArrProducer \
                ImgAlgos.NDArrAverage

[ImgAlgos.EpixNDArrProducer]
source = DetInfo(:Epix10k)
key_in =
key_out = epix-nda
outtype = float
print_bits = 255

[ImgAlgos.NDArrAverage]
source       = DetInfo(:Epix10k)
key          = epix-nda
avefile      = epix-ave
print_bits   = 29
```

`EpixNDArrProducer` gets Epix10k ndarray from data and saves it in the events store with possible type conversion. Then `NDArrAverage` averages this ndarray over requested number of events (100) and saves it in file.

Example for module ImgAlgos::PnccdImageProducer

See Module ImgAlgos::PnccdImageProducer

Example of the psana configuration file:

```
[psana]
files         = /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/<file-name-1>.xtc\
                /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/<file-name-2>.xtc
#skip-events = 100
events      =  5
modules = ImgAlgos.PnccdImageProducer ImgAlgos.ImgSaveInFile

[ImgAlgos.PnccdImageProducer]
source        = DetInfo(:pnCCD)
inkey         =
outimgkey     = imgpnccd
print_bits    = 1

[ImgAlgos.ImgSaveInFile]
source        = DetInfo(:pnCCD)
key           = imgpnccd
fname         = pnccd-img-ev
saveAll       = true
#eventSave     = 82
print_bits    = 1
```

This script saves text files with images like `pnccd-img-ev-<run-date-time.nsec>.txt`, which can be presented as:

Advanced example for PnccdImageProducer;

- get pnccd ndarray from data,
- calibrate ndarray (subtract pedestals, common mode, remove pixels with bad status),
- produce image with two gaps from calibrated ndarray,
- average image for 10 events:

```
[psana]
#calib-dir = /reg/d/psdm/SXR/sxrb5914/calib
files = exp=sxrb5914:run=245
events = 10

modules = ImgAlgos.Tahometer \
          ImgAlgos.PnccdNDArrProducer \
          ImgAlgos.NDArrCalib \
          ImgAlgos.PnccdImageProducer \
          ImgAlgos.NDArrAverage

[ImgAlgos.Tahometer]
dn         = 100
print_bits = 7

[ImgAlgos.PnccdNDArrProducer]
source  = DetInfo(Camp.0:pnCCD.1)
key_in  =
key_out = pnccd-ndarr
outtype = asdata
print_bits = 0

[ImgAlgos.NDArrCalib]
source = DetInfo(Camp.0:pnCCD.1)
key_in = pnccd-ndarr
key_out = calibrated
do_peds = yes
do_cmod = yes
do_stat = yes
do_mask = no
do_bkgd = no
do_gain = no
do_nrms = no
do_thre = no
#fname_mask = pnccd-test-mask.txt
#fname_bkgd = pnccd-test-bkgd.txt
masked_value     =    0
threshold_nrms   =  4.0
threshold        =  100
below_thre_value =    0
bkgd_ind_min     = 10000
bkgd_ind_max     = 10200
bkgd_ind_inc     =    1
print_bits       =    1

[ImgAlgos.PnccdImageProducer]
source        = DetInfo(Camp.0:pnCCD.1)
inkey         = calibrated
outimgkey     = pnccd-img
gap_rows      = 0
gap_cols      = 16
gap_value     = 0
print_bits    = 1

[ImgAlgos.NDArrAverage]
source       = DetInfo(Camp.0:pnCCD.1)
key          = pnccd-img
avefile      = pnccd-ave
rmsfile      = pnccd-rms
#maskfile     = pnccd-msk
#hotpixfile   = pnccd-hot
thr_rms_ADU  = 160
thr_min_ADU  = 2
thr_max_ADU  = 10000
print_bits   = 29
```

Example for module ImgAlgos::CameraImageProducer

See Module ImgAlgos::CameraImageProducer

Example of the psana configuration file:

```
[psana]
files           = /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/<file-name-1>.xtc
modules         = ImgAlgos.CameraImageProducer ImgAlgos.ImgSaveInFile
events          = 5

[ImgAlgos.CameraImageProducer]
source          = DetInfo(:Opal1000)
key_in          =
key_out         = img
subtract_offset = true
print_bits      = 15

[ImgAlgos.ImgSaveInFile]
source        = DetInfo(:Opal1000)
key           = img
fname         = img-from-my-experiment
saveAll       = true
#eventSave     = 1
```

This script saves text files with images like `img-from-my-experiment-<run-date-time.nsec>.txt`, which can be drawn by the python script

> *./ImgAlgos/data/PlotCameraImageFromFile.py <filename>.txt <Amin> <Amax>*

Example for module ImgAlgos::PrincetonImageProducer

See Module ImgAlgos::PrincetonImageProducer

Example of the psana configuration file:

```
[psana]
files         = /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/<file-name-1>.xtc\
                /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/<file-name-2>.xtc
modules       = ImgAlgos.PrincetonImageProducer \
                ImgAlgos.ImgSaveInFile
events          = 3

[ImgAlgos.PrincetonImageProducer]
source          = DetInfo(:Princeton)
key_in          =
key_out         = img
subtract_offset = true
print_bits      = 31

[ImgAlgos.ImgSaveInFile]
source        = DetInfo(:Princeton)
key           = img
fname         = img-xcs
saveAll       = true
print_bits    = 31
```

Example for module ImgAlgos::AcqirisArrProducer

See description of parameters in Module ImgAlgos::AcqirisArrProducer

Example of the psana configuration file `psana-amo01509-r0125-acqiris.cfg` :

```
# Command to run this script:
# psana -c psana-amo01509-r0125-acqiris.cfg;
#
# Useful commands:
# psana -n 5 -m EventKeys exp=amo01509:run=125:xtc > test-acqiris-file.txt;
# psana -n 5 -m psana_examples.DumpAcqiris exp=amo01509:run=125:xtc > test-acqiris-file.txt;    (C++ version)
# psana -n 1 -m psana_examples.dump_acqiris exp=amo01509:run=125:xtc > test-acqiris-file.txt;   (python version)

[psana]
#files = /reg/d/psdm/AMO/amo01509/xtc/e8-r0125-s00-c00.xtc /reg/d/psdm/AMO/amo01509/xtc/e8-r0125-s01-c00.xtc
files = exp=amo01509:run=125:xtc

modules = ImgAlgos.AcqirisArrProducer ImgAlgos.ImgSaveInFile:wf ImgAlgos.ImgSaveInFile:wt

skip-events = 0
events      = 100


[ImgAlgos.AcqirisArrProducer]
source        = AmoETOF.0:Acqiris.0
key_in        =
key_wform     = acqiris_wform
key_wtime     = acqiris_wtime
fname_prefix  = acq
print_bits    = 11


[ImgAlgos.ImgSaveInFile:wf]
source         = AmoETOF.0:Acqiris.0
key            = acqiris_wform
fname          = acq-AmoETOF-wform
ftype          = txt
#ftype          = tiff
#saveAll        = true
print_bits     = 3
eventSave      = 5


[ImgAlgos.ImgSaveInFile:wt]
source         = AmoETOF.0:Acqiris.0
key            = acqiris_wtime
fname          = acq-AmoETOF-wtime
ftype          = txt
#ftype          = tiff
#saveAll        = true
print_bits     = 3
eventSave      = 5
```

This script with psana does a few things:

- module AcqirisArrProducer gets Acqiris data objects from event store, adjusts trigger time corrections, and saves them back in the event store as uniform ndarrays<double,2> objects for waveforms and times
- two instances of the module ImgSaveInFile save arrays of waveforms and wave-times for locally counted event 5.

This script saves 3 text files:

- `acq-amo01509-r0125.txt` -- with configuration parameters:

```
Acqiris::ConfigV1:
  nbrBanks=1 channelMask=69905 nbrChannels=5 nbrConvertersPerChannel=4
  horiz: sampInterval=2.5e-10 delayTime=0 nbrSegments=1 nbrSamples=10000
  vert(0): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(1): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(2): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(3): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(4): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
```

- acq-AmoETOF-wform-r0125-20091018-182512.194787218.txt – with waveforms for 5th event

- acq-AmoETOF-wtime-r0125-20091018-182512.194787218.txt – with wave-times for 5th event

Arrays from these files can be plotted for all channels, shown by different colors:

Example for module ImgAlgos::AcqirisAverage

See description of parameters in Module ImgAlgos::AcqirisAverage

Configuration file `psana-amo01509-r0125-acqiris-average.cfg`:

```
# Command to run this script:
# psana -c psana-amo01509-r0125-acqiris-average.cfg;
#
# Useful commands:
# psana -n 5 -m EventKeys exp=amo01509:run=125:xtc > test-acqiris-file.txt;
# psana -n 5 -m psana_examples.DumpAcqiris exp=amo01509:run=125:xtc > test-acqiris-file.txt;    (C++ version)
# psana -n 1 -m psana_examples.dump_acqiris exp=amo01509:run=125:xtc > test-acqiris-file.txt;   (python version)

[psana]
#files = /reg/d/psdm/AMO/amo01509/xtc/e8-r0125-s00-c00.xtc /reg/d/psdm/AMO/amo01509/xtc/e8-r0125-s01-c00.xtc
files = exp=amo01509:run=125:xtc

modules = ImgAlgos.AcqirisArrProducer ImgAlgos.AcqirisAverage

skip-events = 0
events      = 1000


[ImgAlgos.AcqirisArrProducer]
source        = AmoETOF.0:Acqiris.0
key_in        =
key_wform     = acqiris_wform
key_wtime     = acqiris_wtime
fname_prefix  = acq
print_bits    = 3


[ImgAlgos.AcqirisAverage]
source               = AmoETOF.0:Acqiris.0
key_in               = acqiris_wform
key_ave              = acqiris_average
fname_ave_prefix     = acq
thresholds           = -0.005 -0.005 -0.005 -0.005 -0.005
is_positive_signal   = no
do_inverse_selection = yes
#skip_events          = 0
#proc_events          = 100
print_bits           = 255
```

Psana with this script runs over 1000 events apply threshold-based selection algorithm and produces files:

- acq-amo01509-r0125-config.txt -- with Acqiris configuration parameters:

```
Acqiris::ConfigV1:
  nbrBanks=1 channelMask=69905 nbrChannels=5 nbrConvertersPerChannel=4
  horiz: sampInterval=2.5e-10 delayTime=0 nbrSegments=1 nbrSamples=10000
  vert(0): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(1): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(2): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(3): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(4): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
```

- acq-amo01509-r0125-ave-wfs.txt with averaged array of waveforms, which were below threshold (averaging for baseline level):

Example for module ImgAlgos::AcqirisCalib

See description of parameters in Module ImgAlgos::AcqirisCalib

Configuration file `psana-amo01509-r0125-acqiris-calib.cfg`:

```
# Command to run this script:
# psana -c psana-amo01509-r0125-acqiris-calib.cfg;
#
# Useful commands:
# psana -n 5 -m EventKeys exp=amo01509:run=125:xtc > test-acqiris-file.txt;
# psana -n 5 -m psana_examples.DumpAcqiris exp=amo01509:run=125:xtc > test-acqiris-file.txt;    (C++ version)
# psana -n 1 -m psana_examples.dump_acqiris exp=amo01509:run=125:xtc > test-acqiris-file.txt;   (python version)

[psana]
#files = /reg/d/psdm/AMO/amo01509/xtc/e8-r0125-s00-c00.xtc /reg/d/psdm/AMO/amo01509/xtc/e8-r0125-s01-c00.xtc
files = exp=amo01509:run=125:xtc

modules = ImgAlgos.AcqirisArrProducer ImgAlgos.AcqirisCalib ImgAlgos.ImgSaveInFile:wf_raw ImgAlgos.
ImgSaveInFile:wf_calib

skip-events = 0
events      = 10


[ImgAlgos.AcqirisArrProducer]
source        = AmoETOF.0:Acqiris.0
key_in        =
key_wform     = acqiris_wform
key_wtime     = acqiris_wtime
fname_prefix  = acq
print_bits    = 1


[ImgAlgos.AcqirisCalib]
source               = AmoETOF.0:Acqiris.0
key_in               = acqiris_wform
key_out              = wf-calibrated
fname_base_line      = acq-amo01509-r0125-ave-wfs.txt
#skip_events          = 0
#proc_events          = 100
print_bits           = 255


[ImgAlgos.ImgSaveInFile:wf_raw]
source         = AmoETOF.0:Acqiris.0
key            = acqiris_wform
fname          = acq-AmoETOF-wform-raw
ftype          = txt
#ftype          = tiff
#saveAll        = true
print_bits     = 3
eventSave      = 5


[ImgAlgos.ImgSaveInFile:wf_calib]
source         = AmoETOF.0:Acqiris.0
key            = wf-calibrated
fname          = acq-AmoETOF-wform-calibrated
ftype          = txt
#ftype          = tiff
#saveAll        = true
print_bits     = 3
eventSave      = 5
```

In this script the base-line level for all waveforms is loaded from file and is subtracted in module `ImgAlgos.AcqirisCalib`.

For example, the raw and calibrated waveforms for event 5 were saved in the files:

- `acq-AmoETOF-wform-raw-r0125-e00000005-20091018-182512.194787218.txt`
- `acq-AmoETOF-wform-calibrated-r0125-e00000005-20091018-182512.194787218.txt`

which content is presented on plots:

Example for module ImgAlgos::AcqirisCFD

See description of parameters in Module ImgAlgos::AcqirisCFD

Configuration file `psana_cfd.cfg`:

```
# Command to run this script:
# psana -c psana_cfd.cfg;
#
[psana]
modules = ImgAlgos.AcqirisArrProducer ImgAlgos.AcqirisCalib ImgAlgos.AcqirisCFD
files = /reg/d/psdm/AMO/amo01509/xtc/e8-r0125-s00-c00.xtc
[ImgAlgos.AcqirisArrProducer]
source  = AmoETOF.0:Acqiris.0
key_in  =
key_wform  = acqiris_wform
key_wtime  = acqiris_wtime
fname_prefix  = acq
print_bits  = 0
[ImgAlgos.AcqirisCalib]
source  = AmoETOF.0:Acqiris.0
key_in  = acqiris_wform
key_out  = wf-calibrated
fname_base_line  = acq--r0000-ave-wfs.txt
#skip_events  = 0
#proc_events  = 100
print_bits  = 0
[ImgAlgos.AcqirisCFD]
source  = AmoETOF.0:Acqiris.0
key_wform  = wf-calibrated
baselines =  0.0 0.0 0.0 0.0 0.0
thresholds =  -0.005 -0.005 -0.005 -0.005 -0.005
fractions =  0.5 0.5 0.5 0.5 0.5
deadtimes =  20.0e-9 20.0e-9 20.0e-9 20.0e-9 20.0e-9
leading_edges = 1 1 1 1 1
```

This script analyzes an AMO run where 5 acqiris channels were in use. It uses an AcqirisArrProducer to producer the list of waveforms/times for all channels. It then uses an AcqirisCalib module to do a baseline subtraction using a previously generated file of baselines act--r0000-ave-wfs.txt. Finally, a constant fraction discriminator algorithm is run on all the waveforms with user specified parameters. The edges calculated by AcqirisCFD are saved to the event as one ndarray<double,1> per channel, each with a (default) key "acqiris_edges_N" where N is the channel number. Channels where no edges were found are not saved to the event.

Example for combination of Acqiris modules

See description of parameters in Modules ImgAlgos::AcqirisArrProducer, AcqirisAverage, and AcqirisCalib

Configuration file `psana-amo01509-r0125-acqiris-comb.cfg`:

```
# Command to run this script:
# psana -c psana-amo01509-r0125-acqiris-comb.cfg;

[psana]
files = exp=amo01509:run=125:xtc

modules = ImgAlgos.AcqirisArrProducer ImgAlgos.AcqirisAverage:bl  ImgAlgos.AcqirisCalib ImgAlgos.AcqirisAverage:
signal ImgAlgos.Tahometer

#skip-events = 0
events      = 2010


[ImgAlgos.AcqirisArrProducer]
source        = AmoETOF.0:Acqiris.0
key_in        =
key_wform     = acqiris_wform
key_wtime     = acqiris_wtime
fname_prefix  = acq
print_bits    = 7


[ImgAlgos.AcqirisAverage:bl]
source               = AmoETOF.0:Acqiris.0
key_in               = acqiris_wform
#key_ave              =
fname_ave_prefix     = acq-bline
thresholds           = -0.005 -0.005 -0.005 -0.005 -0.005
is_positive_signal   = no
do_inverse_selection = yes
skip_events          = 0
proc_events          = 1000
print_bits           = 31


[ImgAlgos.AcqirisCalib]
source               = AmoETOF.0:Acqiris.0
key_in               = acqiris_wform
key_out              = wf-calibrated
fname_base_line      = acq-bline
skip_events          = 1001
proc_events          = 1000
print_bits           = 47


[ImgAlgos.AcqirisAverage:signal]
source               = AmoETOF.0:Acqiris.0
key_in               = wf-calibrated
#key_ave              =
fname_ave_prefix     = acq-signal
thresholds           = -0.01 -0.01 -0.01 -0.01 -0.01
is_positive_signal   = no
do_inverse_selection = no
skip_events          = 1001
proc_events          = 1000
print_bits           = 31


[ImgAlgos.Tahometer]
print_bits  = 7
```

This script works with psana as follows:

- for the 1st 1000 events averages waveforms below threshold and saves results in the file acq-bline-amo01509-r0125-ave-wfs.txt;
- for the next 1000 events subtracts baseline level and averages waveforms above thresholds and saves results in the file acq-signal-amo01509-r0125-ave-wfs.txt.

This script produces three files:

- acq-amo01509-r0125-config.txt - with Acqris configuration parameters:

```
Acqiris::ConfigV1:
  nbrBanks=1 channelMask=69905 nbrChannels=5 nbrConvertersPerChannel=4
  horiz: sampInterval=2.5e-10 delayTime=0 nbrSegments=1 nbrSamples=10000
  vert(0): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(1): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(2): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(3): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
  vert(4): fullScale=0.1 slope=1.52588e-06 offset=0 coupling=3 bandwidth=0
```

- acq-bline-amo01509-r0125-ave-wfs.txt - contains baseline averaged arrays, which can be presented by full scale and zoomed plots:

- acq-signal-amo01509-r0125-ave-wfs.txt- contains signal averaged arrays, which can be presented by full scale and zoomed plots:

Example for module ImgAlgos::NDArrImageProducer

See Module ImgAlgos::NDArrImageProducer

Module ImgAlgos.NDArrImageProducer produces image from any detector-associated ndarray

Possible chain of modules:

- <Package>.<Detector>NDArrProducer
- ImgAlgos.NDArrCalib
- ImgAlgos.NDArrImageProducer
- ImgAlgos.NDArrAverage

Example of the configuration file for cspad

```
[psana]
files = exp=cxii0114:run=227
events = 100

modules = ImgAlgos.Tahometer \
          CSPadPixCoords.CSPadNDArrProducer \
          ImgAlgos.NDArrCalib \
          ImgAlgos.NDArrImageProducer \
          ImgAlgos.NDArrAverage

[CSPadPixCoords.CSPadNDArrProducer]
source       = DetInfo(CxiDs1.0:Cspad.0)
inkey        =
outkey       = cspad_ndarr
outtype      = float
is_fullsize  = yes
print_bits   = 3

[ImgAlgos.NDArrCalib]
source  = DetInfo(CxiDs1.0:Cspad.0)
key_in  = cspad_ndarr
key_out = calibrated
do_peds = yes
do_cmod = yes
do_stat = yes
do_mask = no
do_bkgd = no
do_gain = no
do_nrms = no
do_thre = no
fname_mask =
fname_bkgd =
masked_value     =  -10
threshold_nrms   =    4
threshold        =    7
below_thre_value =    0
print_bits       =    3

[ImgAlgos.NDArrImageProducer]
source       = CxiDs1.0:Cspad.0
key_in       = calibrated
key_out      = cspad_img
#type_out     = asinp
#type_out     = float
#x0_off_pix   = 50
#y0_off_pix   = 50
print_bits   = 255
#calibdir     = /reg/neh/home/dubrovin/LCLS/CSPadAlignment-v01/calib-cxi-ds1-2014-03-19/calib

[ImgAlgos.NDArrAverage]
source       = DetInfo(CxiDs1.0:Cspad.0)
key          = cspad_img
avefile      = cspad-img-ave
rmsfile      = cspad-img-rms
#maskfile    = cspad-img-msk
#hotpixfile  = cspad-img-hot
thr_rms_ADU  = 0
thr_min_ADU  = 4
thr_max_ADU  = 65000
print_bits   = 29
```

Example of the configuration file for cspad2x2

```
[psana]
# calib-dir = /reg/d/psdm/mec/meca1113/calib
#calib-dir = /reg/neh/home/dubrovin/LCLS/CSPad2x2Alignment/calib-cspad2x2-01-2013-02-13/calib

files = exp=meca1113:run=376
events = 10
#skip-events = 0

modules = CSPadPixCoords.CSPad2x2NDArrProducer \
          ImgAlgos.NDArrImageProducer \
          ImgAlgos.NDArrAverage

[CSPadPixCoords.CSPad2x2NDArrProducer]
source     = MecTargetChamber.0:Cspad2x2.1
inkey      =
outkey     = cspad2x2_ndarr
outtype    = int16
print_bits = 5

[ImgAlgos.NDArrImageProducer]
source       = MecTargetChamber.0:Cspad2x2.1
key_in       = cspad2x2_ndarr
key_out      = cspad2x2_img
#type_out     = asinp
#type_out     = float
#x0_off_pix   = 50
#y0_off_pix   = 50
print_bits   = 255
#oname        = CSPAD2X2:V1
#oindex       = 0
#pix_scale_size_um = 218.
#calibdir     = /reg/neh/home/dubrovin/LCLS/CSPad2x2Alignment/calib-cspad2x2-01-2013-02-13/calib
#calibgroup   = CsPad2x2::CalibV1


[ImgAlgos.NDArrAverage]
source       = MecTargetChamber.0:Cspad2x2.1
#key          = cspad2x2_ndarr
key          = cspad2x2_img
avefile      = arr-ave
rmsfile      = arr-rms
maskfile     = arr-msk
hotpixfile   = arr-hot
thr_rms_ADU  = 10
#thr_min_ADU = 2
#thr_max_ADU = 20000
print_bits   = 255
```

Example for module ImgAlgos::NDArrAverage

- See Module ImgAlgos::NDArrAverage andModule CSPadPixCoords::CSPadNDArrProducer
- The `NDArrAverage` module in combination with CSPadNDArrProducer (or any other device NDArrProducer) can be used for evaluation of averaged pedestals or background using dedicated runs.
  Typical configuration file may looks like this:

```
# Run this script:
# psana -c psana-meca1113-r0045-cspad-cspad-dark-hotpix.cfg

[psana]
files = exp=meca1113:run=45
events = 400
#skip-events = 0

modules = CSPadPixCoords.CSPadNDArrProducer \
          ImgAlgos.NDArrAverage \
          ImgAlgos.Tahometer


[CSPadPixCoords.CSPadNDArrProducer]
source       = MecTargetChamber.0:Cspad.0
inkey        =
outkey       = cspad_ndarr
outtype      = int16
is_fullsize  = yes
print_bits   = 3


[ImgAlgos.NDArrAverage]
source       = MecTargetChamber.0:Cspad.0
key          = cspad_ndarr
avefile      = cspad.0-ave
rmsfile      = cspad.0-rms
maskfile     = cspad.0-msk
hotpixfile   = cspad.0-hot
thr_rms_ADU  = 10
thr_min_ADU  = 4
thr_max_ADU  = 10000
print_bits   = 29
#evts_stage1  = 100
#gate_width1  = 500.
#evts_stage2  = 200
#gate_width2  = 100.


[ImgAlgos.Tahometer]
dn         = 100
print_bits = 7
```

- Module `ImgAlgos.Tahometer` is not required in this configuration file and is added for convenience to print timing statistics for this job.
- Evaluation of average intensity in 2 or 3 stages using gate-based algorithms excludes out-layers in intensity spectra and makes average more stable and reliable. However, the gate width is not an universal parameter. In order to set this parameter one has to look at spectrum of averaged intensities for particular device. The same is valid for evaluation of hot/bad pixel masks. One has to plot spectra of averaged intensity and rms values. Averaged intensity and rms spectra for exp=meca1113:run=45 are shown on plots, respectively:

Then, one has to decide how to set parameters for NDArrAverage algorithms, for example, it is quite safe to use

- thr_rms_ADU = 10 – to discard very noisy pixels,
- thr_min_ADU = 4 – to discard presumably dead pixels with 0-intensity,
- thr_max_ADU = 10000 – to discard pixels with intensity significantly exceeding average value. To be on safe side for int16 data this parameter can be set to 2^{16}-4, where 4 in both cases is just a small arbitrary number for spare safety gap.

Example for module ImgAlgos::NDArrCalib

- See Module ImgAlgos::NDArrCalib, Module ImgAlgos::NDArrAverageThe `NDArrCalib` module in combination with any device NDArrProducer (f or example PnccdNDArrProducer) can be used to apply intensity corrections to ndarray.
  Typical configuration files are shown below.

Example of ImgAlgos::NDArrCalib for pnCCD

```
[psana]
files = exp=amoa1214:run=7
#skip-events = 100
events      = 5

modules = ImgAlgos.Tahometer \
          ImgAlgos.PnccdNDArrProducer \
          ImgAlgos.NDArrCalib \
          ImgAlgos.PnccdImageProducer \
          ImgAlgos.ImgSaveInFile
#         ImgAlgos.NDArrAverage \


[ImgAlgos.PnccdNDArrProducer]
source        = DetInfo(Camp.0:pnCCD.0)
key_in  =
key_out = pnccd-ndarr
outtype = asdata
print_bits = 13


[ImgAlgos.NDArrCalib]
source = DetInfo(Camp.0:pnCCD.0)
key_in = pnccd-ndarr
key_out = calibrated
outtype = float
do_peds = yes
do_cmod = yes
do_stat = no
do_mask = no
do_bkgd = no
do_gain = no
do_nrms = no
do_thre = no
fname_mask =
fname_bkgd =
masked_value     =    0
threshold_nrms   =    3
threshold        =  100
below_thre_value =    0
bkgd_ind_min     =    0
bkgd_ind_max     = 1000
bkgd_ind_inc     =   10
print_bits       =  255


[ImgAlgos.PnccdImageProducer]
source        = DetInfo(Camp.0:pnCCD.0)
#inkey         = pnccd-ndarr
inkey         = calibrated
outimgkey     = pnccd-img
gap_size      = 20
gap_value     = 0
print_bits    = 1


[ImgAlgos.ImgSaveInFile]
source        = DetInfo(Camp.0:pnCCD.0)
key           = pnccd-img
fname         = pnccd-img-from-arr
ftype         = txt
saveAll       = true
print_bits    = 31



[ImgAlgos.Tahometer]
dn         = 100
print_bits = 7
```

Example of ImgAlgos::NDArrCalib for CSPAD

```
[psana]
files = exp=cxi83714:run=136
events = 100

modules = ImgAlgos.Tahometer \
          CSPadPixCoords.CSPadNDArrProducer \
          ImgAlgos.NDArrCalib \
          ImgAlgos.NDArrAverage

[CSPadPixCoords.CSPadNDArrProducer]
source       = DetInfo(CxiDs1.0:Cspad.0)
inkey        =
outkey       = cspad_ndarr
outtype      = float
is_fullsize  = yes
print_bits   = 3

[ImgAlgos.NDArrCalib]
source  = DetInfo(CxiDs1.0:Cspad.0)
key_in  = cspad_ndarr
key_out = calibrated
outtype = double
do_peds = yes
do_cmod = yes
do_stat = yes
do_mask = no
do_bkgd = no
do_gain = no
do_nrms = no
do_thre = no
fname_mask =
fname_bkgd =
masked_value     =  -10
threshold_nrms   =    4
threshold        =    7
below_thre_value =    0
bkgd_ind_min     =    0
bkgd_ind_max     = 1000
bkgd_ind_inc     =   10
print_bits       =    3

[ImgAlgos.NDArrAverage]
source       = DetInfo(CxiDs1.0:Cspad.0)
key          = calibrated
avefile      = cspad-ave
rmsfile      = cspad-rms
#maskfile     = cspad-msk
#hotpixfile   = cspad-hot
thr_rms_ADU  = 10
thr_min_ADU  = 4
thr_max_ADU  = 10000
print_bits   = 29

[ImgAlgos.Tahometer]
dn         = 10
print_bits = 7
```

For test purpose we use exp=cxi83714:run=136 and loop over 100 exents.

Case 1: raw, non-calibrated images

do_peds = no; 50-60 ms/event

plots for average and rms value distribution for all pixels

Case 2: images with subtracted pedestals and applied status mask

do_peds = yes; 60-70 ms/event
do_stat = yes

Case 3: The same as 2 with common mode subtracted

do_cmod = yes; 107-117 ms/event

Andy's algorithm for CSPAD common mode correction with minor adaptive modifications is applied with parameters:

/reg/d/psdm/cxi/cxi83714/calib/CsPad::CalibV1/CxiDs1.0:Cspad.0/common_mode/135-136.data

1 10 10 100 - algorithm mode, allowed peak mean, allowed peak rms, threshold on number of pixels in ADU bin.

Comparison with Andy's module `cspad_mod.CsPadCalib` for common mode correction of int16_t data

```
[psana]
# Default calibration directory:
# calib-dir = /reg/d/psdm/mec/cxi83714/calib
# calib-dir = /reg/neh/home1/dubrovin/LCLS/CSPadAlignment-v01/calib-mec-2013-12-10/

files = exp=cxi83714:run=136
events = 100
#skip-events = 4000

modules = cspad_mod.CsPadCalib \
          CSPadPixCoords.CSPadNDArrProducer \
          ImgAlgos.NDArrAverage \
          ImgAlgos.Tahometer

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated_data
doPedestals   = yes
doPixelStatus = yes
doCommonMode  = yes

[CSPadPixCoords.CSPadNDArrProducer]
source       = DetInfo(CxiDs1.0:Cspad.0)
inkey        = calibrated_data
outkey       = cspad_ndarr
outtype      = float
is_fullsize  = yes
print_bits   = 3

[ImgAlgos.NDArrAverage]
source       = DetInfo(CxiDs1.0:Cspad.0)
key          = cspad_ndarr
avefile      = cspad-calib-ave
rmsfile      = cspad-calib-rms
#maskfile     = cspad-msk
#hotpixfile   = cspad-hot
thr_rms_ADU  = 10
thr_min_ADU  = 4
thr_max_ADU  = 10000
print_bits   = 29

[ImgAlgos.Tahometer]
dn         = 10
print_bits = 7
```

doPedestals = yes

doCommonMode = no; 100-110ms

doPedestals = yes
doPixelStatus = yes
doCommonMode = yes; 150-160ms

Conclusion:

- Results of `ImgAlgos.NDArrCalib` well reproduce `cspad_mod.CsPadCalib`
- Common mode correction shrinks the width of averaged intensities from 0.60 to 0.54, and rms from 3.78 to 3.27.
- Algorithm is quite time expensive, it takes 30-40ms/event.

Example of ImgAlgos::NDArrCalib for Fccd960

- **Calibration**

To calibrate fccd960 for dark runs use command

> *calibman*

For test purpose a couple of dark runs were processed and constants were deployed:

```
Content of: /reg/d/psdm/XCS/xcsd7814/calib for detector: Fccd960
    Camera::CalibV1
        XcsEndstation.0:Fccd960.0
            pixel_rms
                12-13.data      file is not used
                78-81.data      run range 0078 - 0081
            pedestals
                12-13.data      file is not used
                78-81.data      run range 0078 - 0081
            pixel_status
                12-13.data      file is not used
                78-81.data      run range 0078 - 0081
```

- **Access data in psana**

Example of the configuration file for psana psana-xcsd7814-r0079-fccd960-aver.cfg can be processed by the command

> *psana -c psana-xcsd7814-r0079-fccd960-aver.cfg*

Two averaged images are saved in the text files for exp=xcsd7814:run=79 for raw and calibrated (subtracted dark level) data.

Gain bit coded gain factor 8 is applied automatically.

? Unknown Attachment ? Unknown Attachment

Example for module ImgAlgos::NDArrDropletFinder

See description in psana - Module Catalog and examples in Peak Finding Module

Example for module ImgAlgos::PixCoordsProducer

See Module ImgAlgos::PixCoordsProducer

Psana configuration file which produces enough data to get CSPAD calibrated intensity and coordinate arrays:

```
[psana]
modules         = ImgAlgos.PixCoordsProducer \
          cspad_mod.CsPadCalib \
          CSPadPixCoords.CSPadNDArrProducer

[ImgAlgos.PixCoordsProducer]
source     = CxiDs1.0:Cspad.0
print_bits = 0

[cspad_mod.CsPadCalib]
source          = CxiDs1.0:Cspad.0
inputKey        =
outputKey       = calibrated
doPedestals     = yes
doPixelStatus   = no
doCommonMode    = no

[CSPadPixCoords.CSPadNDArrProducer]
source          = CxiDs1.0:Cspad.0
inkey           = calibrated
outkey          = cspad_ndarr
outtype         = int16
is_fullsize     = yes
print_bits      = 3
```

Additional keywors need to be added to the list of module parameters in order to evaluate pixel area and coordinate indexes (for image) arrays. For example:

```
[ImgAlgos.PixCoordsProducer]
source       = CxiDs1.0:Cspad.0
key_out_area = pix_area
key_out_mask = pix_mask
key_out_ix   = pix_ix
key_out_iy   = pix_iy
x0_off_pix   = 1000
y0_off_pix   = 1000
mask_bits    = 15
print_bits   = 255
```

In python code these arrays can be obtained with env.calibStore().get(...) method:

```
    env = ds.env()
    cls = env.calibStore()
    src = psana.Source('DetInfo(CxiDs1.0:Cspad.0)')
    ...
    A = evt.get(psana.ndarray_int16_3,   src, 'cspad_ndarr').flatten()

    X    = cls.get(psana.ndarray_float64_1, src, 'x-pix-coords')
    Y    = cls.get(psana.ndarray_float64_1, src, 'y-pix-coords')
    Area = cls.get(psana.ndarray_float64_1, src, 'pix_area')
    Mask = cls.get(psana.ndarray_int32_1,   src, 'pix_mask')
    iX   = cls.get(psana.ndarray_uint32_1,  src, 'pix_ix')
    iY   = cls.get(psana.ndarray_uint32_1,  src, 'pix_iy')
    fname= cls.get(str,                     src, 'geometry-fname')

    // depricated method since ana-0.13.3:
    path_nda = cls.get(psana.ndarray_uint8_1, src, 'geometry-calib')
    path = ''.join(map(chr, path_nda)) if path_nda is not None else 'N/A'
```

Their shape=(32*185*388,) = (2296960,)

These arrays can be processed by users' code directly. In particular, it is easy to conver them in 2-d image numpy array and plot it:

```
import numpy as np
import matplotlib.pyplot as plt
from PSCalib.GeometryAccess import img_from_pixel_arrays

    pix_size = 109.92
    xmin, ymin = X.min()-pix_size/2, Y.min()-pix_size/2
    iX, iY = np.array((X-xmin)/pix_size, dtype=np.uint), np.array((Y-ymin)/pix_size, dtype=np.uint)
    img = img_from_pixel_arrays(iX,iY,W=A)

    plt.imshow(img)
    plt.draw()
```

Example for module ImgAlgos::ImgAverage

- See Module ImgAlgos::ImgAverage
- The `ImgAverage` module can be used for evaluation of averaged pedestals or background using dedicated runs. Typical configuration file may looks like this:

```
[psana]
files   = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name>.xtc
modules = ImgAlgos.CameraImageProducer \
          ImgAlgos.ImgAverage
events  = 1000

[ImgAlgos.CameraImageProducer]
source          = DetInfo(:Opal1000)
key_in          =
key_out         = img
subtract_offset = true
print_bits      = 1

[ImgAlgos.ImgAverage]
source          = DetInfo(:Opal1000)
key             = img
avefile         = img-ave.dat
rmsfile         = img-rms.dat
print_bits      = 31
evts_stage1     = 100
evts_stage2     = 100
gate_width1     = 200
gate_width2     = 20
```

Example for module ImgAlgos::ImgMaskEvaluation

- See Module ImgAlgos::ImgMaskEvaluation
- Configuration parameters for psana:

```
[ImgAlgos.ImgMaskEvaluation]
source          = DetInfo(:Opal1000)
key             = img
file_mask_satu        = img-mask-satu.dat
file_mask_nois        = img-mask-nois.dat
file_mask_comb        = img-mask-comb.dat
file_frac_satu        = img-frac-satu.dat
file_frac_nois        = img-frac-nois.dat
thre_satu           = 400
frac_satu       = 0
dr              = 1
thre_SoN        = 3
frac_nois       = 0.05
print_bits      = 63
```

⚠ In this example parameters were chosen in order to get a small number of "noisy" pixel just due to statistics.

Typical image:

Plots for fraction of noisy pixels:

Plots for fraction of saturated pixels:

Masks: **noisy**, **saturated**, and **combined**:

...

Example for module ImgAlgos::ImgCalib

See Module ImgAlgos::ImgCalib

```
[ImgAlgos.CameraImageProducer]
source          = DetInfo(:Opal1000)
key_in          =
key_out         = img
subtract_offset = true
print_bits      = 1

[ImgAlgos.ImgCalib]
source          = DetInfo(:Opal1000)
key_in          = img
key_out         = calibrated
fname_peds      = <pedestal-file-name>
fname_mask      = <mask-file-name>
fname_bkgd      =
fname_gain      =
print_bits      = 31
```

Example of the mask file and resulting image:

-

Example for module ImgAlgos::ImgRadialCorrection

See Module ImgAlgos::ImgRadialCorrection

```
[psana]
files = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-1>.xtc \
        /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-2>.xtc \
        ...
        /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-N>.xtc
skip-events = 500
events      = 10
modules = cspad_mod.CsPadCalib \
#         ImgAlgos.CSPadBkgdSubtract \
          CSPadPixCoords.CSPadImageProducer \
          ImgAlgos.ImgRadialCorrection \
          ImgAlgos.ImgSaveInFile:1


[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = no
doCommonMode  = no

[ImgAlgos.CSPadBkgdSubtract]
source        = DetInfo(CxiDs1.0:Cspad.0)
inputKey      = calibrated
outputKey     = bkgd_subtracted_arr
bkgd_fname    = <the-file-name-with-background-array>
norm_sector   = 0
print_bits    = 0

[CSPadPixCoords.CSPadImageProducer]
calibDir      = /reg/d/psdm/<instrument>/<experiment>/calib
typeGroupName = CsPad::CalibV1
source        = CxiDs1.0:Cspad.0
key           = calibrated
imgkey        = current_img
#tiltIsApplied = true

[ImgAlgos.ImgRadialCorrection]
source        = DetInfo(CxiDs1.0:Cspad.0)
inkey         = current_img
outkey        = r_cor_img
xcenter       = 866
ycenter       = 857
rmin          = 100
rmax          = 810
n_phi_bins    = 60
event         = 0
print_bits    = 3

[ImgAlgos.ImgSaveInFile:1]
source        = CxiDs1.0:Cspad.0
key           = r_cor_img
fname         = <file-name-for-image-array>
#saveAll       = true
eventSave     = 8
```

Note: the option of the background subtraction (`ImgAlgos.CSPadBkgdSubtract`) is commented out in this configuration file . In order to evoke this option, the comment sign (#) should be removed from the list of `modules` and the `key=bkgd_subtracted_arr` should be used in `CSPadPixCoords.CSPadImageProducer`.

Calibrated image and spectrum:

Calibrated and radial-corrected image, spectrum, and subtracted r-phi65 distribution for `n_phi_bins=65`:

Calibrated and radial-corrected image, spectrum, and subtracted r-phi12 distribution for `n_phi_bins=12`:

Example for module ImgAlgos::ImgPeakFinder

See Module ImgAlgos::ImgPeakFinder

Configuration file example:

```
[psana]
files         = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-1>.xtc \
                /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-2>.xtc \
                ...
                /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-N>.xtc

modules       = ImgAlgos.ImgPeakFinder PrintSeparator

#skip-events  = 500
events        = 120

[ImgAlgos.ImgPeakFinder]
source         = DetInfo(:Opal1000)
key            =
peaksKey       =  peaks
threshold_low  =  20
threshold_high =  50
sigma          =  1.5
smear_radius   =  2
peak_radius    =  3
xmin           =  200
xmax           =  800
ymin           =  100
ymax           =  900
testEvent      =  115
print_bits     =  0
finderIsOn     =  true
```

This algorithm consumes ~15 ms/event on psana0101 for full Opal1000 (1024x1024) camera image.

Smearing algorithm use a "safety margin" which is currently set to 10 pixels (offset from each boarder of the full image size).

Image on different stages of this algorithm:

raw image,

image in the window with amplitudes above the `threshold_low`

- few peaks at the edges were discarded by the window limits,
- image still contains many 1-photon pixels, which need to be eliminated,

smeared image,

raw image with found peaks (marked by the red circles)

zoom of the previous plot.

Example for module ImgAlgos::ImgPeakFilter

```
[ImgAlgos.ImgPeakFilter]
source         = DetInfo(:Opal1000)
key            = peaks
threshold_peak = 5
threshold_total= 0
n_peaks_min    = 10
print_bits     = 11
fname          = img
selection_mode = SELECTION_ON
```

Example for module ImgAlgos::ImgPeakFinderAB

See Module ImgAlgos::ImgPeakFinderAB

Configuration file example:

```
[psana]
files   = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name>.xtc
modules = ImgAlgos.CameraImageProducer \
          ImgAlgos.ImgPeakFinderAB
events  = 10

[ImgAlgos.CameraImageProducer]
source          = DetInfo(:Opal1000)
key_in          =
key_out         = img
subtract_offset = true
print_bits      = 1

[ImgAlgos.ImgPeakFinderAB]
source                 = DetInfo(:Opal1000)
key                    = img
key_peaks_out          = peaks
#key_signal_out        = signal-arr
#hot_pix_mask_inp_file = ana-misc-exp/mask.dat
#hot_pix_mask_out_file = noise-mask-out.dat
#frac_noisy_evts_file  = noise-frac-out.dat
evt_file_out           = tmp/img-

rmin                =    10
dr                  =     1
SoNThr_noise        =     3
SoNThr_signal       =     3
frac_noisy_imgs     =   0.9
peak_npix_min       =     3
peak_npix_max       =   100
peak_amp_tot_thr    =     0.
peak_SoN_thr        =     4.
event_npeak_min     =     5
event_npeak_max     =  1000
event_amp_tot_thr   =     0.
nevents_mask_update =     0
nevents_mask_accum  =    50
selection_mode      = SELECTION_ON
out_file_bits       =    15
print_bits          =   513
```

Results:

Example for module ImgAlgos::ImgHitFinder

See Module ImgAlgos::ImgHitFinder

ImgHitFinder in regular mode needs in file with pedestals (offset) to correct the image and file with threshold.

In amo74213 run 93 these files can be obtained directly from data, discarding signal hits as outliers using

ImgAlgos.ImgAverage module as follows with configuration file:

```
# File: psana-amo74213-r0093-opal-img-average.cfg

[psana]
#files = /reg/d/psdm/AMO/amo74213/xtc/e269-r0093-s05-c00.xtc
files = exp=amo74213:run=93:xtc

modules = ImgAlgos.CameraImageProducer \
          ImgAlgos.ImgAverage
skip-events = 0
events      = 1000

[ImgAlgos.CameraImageProducer]
source          = DetInfo(:Opal1000)
key_in          =
key_out         = img
subtract_offset = true
print_bits      = 1

[ImgAlgos.ImgAverage]
source          = DetInfo(:Opal1000)
key             = img
avefile         = img-ave-for-peds
rmsfile         = img-rms-for-thre
evts_stage1     = 100
evts_stage2     = 100
gate_width1     = 50
gate_width2     = 10
print_bits      = 31
```

Run psana using command:

```
psana -c psana-amo74213-r0093-opal-img-average.cfg
```

At the end of this procedure two files will be created:

img-ave-for-peds-r0093.dat - may be used for pedestal subtraction:

-

img-rms-for-thre-r0093.dat - may be used multiplied by number of rms as a threshold:

-

The file with accumulated pixel hits can be obtained using configuration file:

```
# File: psana-amo74213-r0093-opal-img-hit-finder.cfg

[psana]
#files = /reg/d/psdm/AMO/amo74213/xtc/e269-r0093-s05-c00.xtc
files = exp=amo74213:run=93:xtc

modules = ImgAlgos.CameraImageProducer \
          ImgAlgos.ImgHitFinder \
          ImgAlgos.ImgAverage
skip-events = 0
events      = 1000

[ImgAlgos.CameraImageProducer]
source          = DetInfo(:Opal1000)
key_in          =
key_out         = img
subtract_offset = true
print_bits      = 1

[ImgAlgos.ImgHitFinder]
source           = DetInfo(:Opal1000)
key_in           = img
key_out          = img_hits
fname_peds       = img-ave-for-peds-r0093.dat
fname_mask       =
fname_gain       =
fname_thre       = img-rms-for-thre-r0093.dat
masked_value     = 0
thre_mode        = 3
#thre_mode        = 2
thre_param       = 5
thre_below_value = 0
thre_above_value = 1
win_row_min      = 10
win_row_max      = 1000
win_col_min      = 10
win_col_max      = 1000
print_bits       = 39

[ImgAlgos.ImgAverage]
source          = DetInfo(:Opal1000)
key             = img_hits
sumfile         = img-sum-result
print_bits      = 25
```

and run it by the command:

```
psana -c psana-amo74213-r0093-opal-img-hit-finder.cfg
```

Which creates the file: img-sum-result-r0093.dat for `thre_mode = 2` and `thre_mode = 3`, respectively:

Example for module ImgAlgos::ImgSpectra

See Module ImgAlgos::ImgSpectra

Configuration file for psana:

```
[psana]
files = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name>.xtc

modules       =  ImgAlgos.CameraImageProducer \
                 ImgAlgos.ImgSpectra \
                 ImgAlgos.ImgSaveInFile:2
#                ImgAlgos.ImgSaveInFile
#                psana_examples.DumpOpal1k \

events          = 100

[ImgAlgos.CameraImageProducer]
source          = DetInfo(:Opal1000)
key_in          =
key_out         = img
subtract_offset = true
print_bits      = 1

[ImgAlgos.ImgSpectra]
source          = DetInfo(:Opal1000)
key_in          = img
key_out         = spectra
sig_band_rowc   = 512.
ref_band_rowc   = 552.
sig_band_tilt   =   0.
ref_band_tilt   =   0.
sig_band_width  =  10
ref_band_width  =  10
print_bits      =  3

[ImgAlgos.ImgSpectraProc]
source          = DetInfo(:Opal1000)
key_in          = spectra
print_bits      =  15

[ImgAlgos.ImgSaveInFile:2]
source        = DetInfo(:Opal1000)
key           = spectra
fname         = spec-xppi0412
saveAll       = true

[ImgAlgos.ImgSaveInFile]
source        = DetInfo(:Opal1000)
key           = img
fname         = img-xppi0412
saveAll       = true
```

This script can be used in order to produce text files with image and spectral array:

or graphics for several images:

Example for module ImgAlgos::ImgSpectraProc

See Module ImgAlgos::ImgSpectraProc

Configuration file for psana:

```
[ImgAlgos.ImgSpectraProc]
source          = DetInfo(:Opal1000)
key_in          = spectra
print_bits      =  15
```

For each event it prints something similar to:

```
[info:ImgAlgos.ImgSpectraProc] Spectral array shape =3, 1024
[info:ImgAlgos.ImgSpectraProc] Image spectra for run=0060 Evt=000100
Column:       0     100     200     300     400     500     600     700     800     900    1000
Signal:    1211    4062   11150   17070   16406   12949    7991    5168    3968    3542    3811
Refer.:     933    3485   10425   17128   17791   13522    8315    5000    3390    2967    3193
Diff. :   0.259   0.153   0.067  -0.003  -0.081  -0.043  -0.040   0.033   0.157   0.177   0.176
[info:ImgAlgos.ImgSpectraProc] Run=0060 Evt=000100 Time=20120507-125420.982421325 done...
```

Example for module ImgAlgos::ImgSaveInFile

```
modules = ... ImgAlgos.ImgSaveInFile:1 ...

[ImgAlgos.ImgSaveInFile:1]
source         = DetInfo(:Opal1000)   # or CxiDs1.0:Cspad.0
key            = img
fname          = my-img
#ftype          = txt
#ftype          = bin
#ftype          = png
ftype          = tiff
#eventSave     = 5
saveAll        = true
```

See Module ImgAlgos::ImgSaveInFile

Example for module ImgPeakFinder and ImgPeakFilter for CSPad

Module `ImgAlgos::ImgPeakFinder` works on image. In order to apply this algorithm to CSPad the image should be produced. In next example the image is produced using consequtive modules `cspad_mod.CsPadCalib`, `ImgAlgos.CSPadMaskApply`, and `CSPadPixCoords.CSPadImageProducer`:

```
[psana]
files   = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-1>.xtc \
          ...
          /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-N>.xtc

events      =  1000
experiment  = cxii0212
calib-dir   = ana-cxii0212/calib

modules = cspad_mod.CsPadCalib \
          ImgAlgos.CSPadMaskApply \
          CSPadPixCoords.CSPadImageProducer \
          ImgAlgos.ImgPeakFinder \
          ImgAlgos.ImgPeakFilter \
          ImgAlgos.ImgSaveInFile:1

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = no
doCommonMode  = yes

[ImgAlgos.CSPadMaskApply]
source            = DetInfo(CxiDs1.0:Cspad.0)
inkey             = calibrated
outkey            = masked_arr
mask_fname        = <your-local-directory>/<mask-file-name>.dat
masked_amp        =  0
print_bits        =  5
mask_control_bits = 15

[CSPadPixCoords.CSPadImageProducer]
calibDir       = /reg/d/psdm/<instrument>/<experiment>/calib
```

```
typeGroupName  = CsPad::CalibV1
source         = CxiDs1.0:Cspad.0
key            = masked_arr
imgkey         = img
print_bits     =  0
#tiltIsApplied  = true

[ImgAlgos.ImgPeakFinder]
source         = DetInfo(CxiDs1.0:Cspad.0)
key            = img
peaksKey       = peaks
threshold_low  = 2
threshold_high = 5
sigma          = 1.5
smear_radius   = 5
peak_radius    = 7
xmin           = 20
xmax           = 1700
ymin           = 20
ymax           = 1700
#testEvent      = 5
print_bits     = 3
#finderIsOn     = true

[ImgAlgos.ImgPeakFilter]
source         = DetInfo(CxiDs1.0:Cspad.0)
key            = peaks
threshold_peak = 5
threshold_total= 0
n_peaks_min    = 10
print_bits     = 11
fname          = cspad-img
selection_mode = SELECTION_ON

[ImgAlgos.ImgSaveInFile:1]
source         = CxiDs1.0:Cspad.0
key            = img
fname          = cspad-img
#eventSave     = 1
saveAll        = true
```

Example for module ImgAlgos::CSPadArrAverage

See Module ImgAlgos::CSPadArrAverage

Configuration file example for evaluation of pedestals:

```
[psana]
modules = ImgAlgos.CSPadArrAverage
files   = <path-to-the-dark-run-file>.xtc

[ImgAlgos.CSPadArrAverage]
source  = DetInfo(CxiDs1.0:Cspad.0)
key     =
avefile = cspad-pedestals-ave.dat
rmsfile = cspad-pedestals-rms.dat
print_bits  = 15
evts_stage1 = 100
evts_stage2 = 100
gate_width1 = 100
gate_width2 =  10
```

Configuration file example for evaluation of background:

```
[psana]
files       = <path-to-the-background-run-file>.xtc
modules     = cspad_mod.CsPadCalib ImgAlgos.CSPadArrAverage
skip-events = 500
events      = 1000000

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = no
doCommonMode  = no

[ImgAlgos.CSPadArrAverage]
source  = DetInfo(CxiDs1.0:Cspad.0)
key     = calibrated
avefile = cspad-background-ave.dat
rmsfile = cspad-background-rms.dat
print_bits  = 15
```

Images of the CSPad arrays for averaged and rms values, respectively, in one of the CXI runs:

Example for module ImgAlgos::CSPadBkgdSubtract

See Module ImgAlgos::CSPadBkgdSubtract

```
[psana]
files = /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-1>.xtc \
        /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-2>.xtc \
        ...
        /reg/d/psdm/<instrument>/<experiment>/xtc/<file-name-N>.xtc
skip-events = 500
events      = 10
modules     = cspad_mod.CsPadCalib ImgAlgos.CSPadBkgdSubtract

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = no
doCommonMode  = no


[ImgAlgos.CSPadBkgdSubtract]
source        = DetInfo(CxiDs1.0:Cspad.0)
inputKey      = calibrated
outputKey     = bkgd_subtracted
bkgd_fname    = <the-file-name-with-background-array>
norm_sector   = 0
print_bits    = 3
```

The file with the background array, `bkgd_fname`, was obtained by averaging 1000 events using module CSPadArrAverage. Subtraction is done with normalization for `norm_sector=0`.

Event image and pixel amplitude spectrum before and after the background subtraction are shown in plots:

Other event with better subtracted background:

Example for Module ImgAlgos::CSPadMaskApply

See Module ImgAlgos::CSPadMaskApply

The array for mask contains zeros and ones for masked and passed pixels, respectively, and has a shape of full-size CSPad array 4*8*185388.

For example, it can be generated by the command

```
./MakePixelMask.py <input-background-cspad-arr-file-name> <threshold> <output-file-name>
```

for the averaged background amplitude array `<input-background-cspad-arr-file-name>` abtained as a result of `ImgAlgos::CSPadArrAverage` module.

Plots show the averaged background, and the mask arrays generated from this background for three thresholds 10, 20, and 30 EDU:

The best results in filtering can be achieved in combination of modiles:

```
modules = cspad_mod.CsPadCalib \
          ImgAlgos.CSPadBkgdSubtract \
          ImgAlgos.CSPadMaskApply \
          ...
[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated_arr
doPedestals   = yes
doPixelStatus = no
doCommonMode  = no

[ImgAlgos.CSPadBkgdSubtract]
source        = DetInfo(CxiDs1.0:Cspad.0)
inputKey      = calibrated_arr
outputKey     = bkgd_subtracted_arr
bkgd_fname    = ana-cxi49012/cspad-cxi49012-r0025-background-ave.dat
norm_sector   = 0
print_bits    = 0

[ImgAlgos.CSPadMaskApply]
source            = DetInfo(CxiDs1.0:Cspad.0)
inkey             = bkgd_subtracted_arr
outkey            = masked_arr
mask_fname        = ana-cxi49012/cspad-cxi49012-r0025-mask-40.dat
masked_amp        =  0
print_bits        =  3
mask_control_bits =  1
```

where

`cspad_mod.CsPadCalib` - subtracts the pedestals from raw CSPad data,

`ImgAlgos.CSPadBkgdSubtract` - subtracts the background,

`ImgAlgos.CSPadMaskApply` - apply the mask.

In the test with images for background represented by the water and solvent rings this filter provides the background suppression factor about 100.

The background images that still pass this filter have significantly larger intensity with respect to averaged background:

Input parameter `mask_control_bits` allows to control masking regions of 2x1. For example, if all edges need to be masked, then use `mask_control_bits = 15`, which gives image array like:

where red regions/lines of pixels of amplitude=8 are masked.

Example for module ImgAlgos::CSPadArrNoise

See Module ImgAlgos::CSPadArrNoise

```
[psana]
modules = cspad_mod.CsPadCalib ImgAlgos.CSPadArrNoise

files   = /reg/d/psdm/cxi/cxi49012/xtc/e158-r0020-s00-c00.xtc \
          /reg/d/psdm/cxi/cxi49012/xtc/e158-r0020-s01-c00.xtc \
          /reg/d/psdm/cxi/cxi49012/xtc/e158-r0020-s02-c00.xtc \
          /reg/d/psdm/cxi/cxi49012/xtc/e158-r0020-s03-c00.xtc \
          /reg/d/psdm/cxi/cxi49012/xtc/e158-r0020-s04-c00.xtc \
          /reg/d/psdm/cxi/cxi49012/xtc/e158-r0020-s05-c00.xtc

#skip-events = 1000
events      = 10

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = no
doCommonMode  = no

[ImgAlgos.CSPadArrNoise]
source          = DetInfo(CxiDs1.0:Cspad.0)
key             = calibrated
statusfile      = ana-cxi49012/cspad-cxi49012-r0020-noise-status.dat
maskfile        = ana-cxi49012/cspad-cxi49012-r0200-noise-mask.dat
print_bits      = 255
rmin            = 3
dr              = 1
SoNThr          = 3
frac_noisy_imgs = 0.15
```

Index map in median algorithm for rmin=3, dr=1:

```
CSPadArrNoise::printMatrixOfIndexesForMedian():
 0 0 0 0 1 0 0 0 0
 0 0 1 1 1 1 1 0 0
 0 1 0 0 0 0 0 1 0
 0 1 0 0 0 0 0 1 0
 1 1 0 0 + 0 0 1 1
 0 1 0 0 0 0 0 1 0
 0 1 0 0 0 0 0 1 0
 0 0 1 1 1 1 1 0 0
 0 0 0 0 1 0 0 0 0
```

Pixel status (fraction of events where `S/N > SoNThr`):

For cspad-cxi49012-r0020 with parameters from confguration file (`frac_noisy_imgs=0.15`) we get, depending on number of events:

`Nnoisy, Ntotal, Nnoisy/Ntotal pixels =94585 2296960 0.041` for 10 events

`Nnoisy, Ntotal, Nnoisy/Ntotal pixels =2112 2296960 0.00092` for 100 events

Pixel mask for noisy pixels with `|S/N| > SoNThr`:

Example for Module ImgAlgos::CSPadArrPeakFinder

See Module ImgAlgos::CSPadArrPeakFinder

```
[psana]
files   = \
   /reg/d/psdm/cxi/cxi49012/xtc/e158-r0150-s00-c00.xtc \
   /reg/d/psdm/cxi/cxi49012/xtc/e158-r0150-s01-c00.xtc \
   /reg/d/psdm/cxi/cxi49012/xtc/e158-r0150-s02-c00.xtc \
   /reg/d/psdm/cxi/cxi49012/xtc/e158-r0150-s03-c00.xtc \
#   /reg/d/psdm/cxi/cxi49012/xtc/e158-r0150-s04-c00.xtc \ suddenly it became unavailable...
   /reg/d/psdm/cxi/cxi49012/xtc/e158-r0150-s05-c00.xtc

#skip-events = 1000
#events      = 200

modules = cspad_mod.CsPadCalib \
          ImgAlgos.CSPadMaskApply \
          ImgAlgos.CSPadArrPeakFinder

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = no
doCommonMode  = no

[ImgAlgos.CSPadMaskApply]
source            = DetInfo(CxiDs1.0:Cspad.0)
inkey             = calibrated
outkey            = masked_arr
mask_fname        = ana-cxi49012/cspad-cxi49012-r0150-mask-badregs.dat
#mask_fname        = ana-cxi49012/cspad-cxi49012-r0150-mask-bkgd.dat
#mask_fname        = ana-cxi49012/cspad-cxi49012-r0150-mask-rects.dat
masked_amp        =  8
print_bits        =  1
mask_control_bits = 15

[ImgAlgos.CSPadArrPeakFinder]
source            = DetInfo(CxiDs1.0:Cspad.0)
key               = masked_arr
key_peaks_out     = peaks

hot_pix_mask_inp_file = ana-cxi49012/cspad-cxi49012-r0150-noise-mask.dat
hot_pix_mask_out_file = ana-cxi49012/cspad-cxi49012-r0150-noise-mask-out.dat
frac_noisy_evts_file  = ana-cxi49012/cspad-cxi49012-r0150-noise-frac.dat

evt_file_out      = tmp/cspad-ev-

rmin              =     3
dr                =     1
SoNThr            =     3
frac_noisy_imgs   =   0.1

peak_npix_min     =     4
peak_npix_max     =    25
peak_amp_tot_thr  =   100.

event_npeak_min   =    10
event_amp_tot_thr =  1000.

nevents_mask_update = 100
nevents_mask_accum  =  50

selection_mode    = SELECTION_ON
out_file_bits     =    15
print_bits        =    512
```

Results:

```
[info:TimeInterval::startTime] Start time: 2012-06-12 15:32:02
[info:ImgAlgos.CSPadArrPeakFinder]   N processed events = 1000  N selected = 55  Fraction of selected = 0.055
[info:ImgAlgos.CSPadArrPeakFinder]   N processed events = 2000  N selected = 62  Fraction of selected = 0.031
[info:ImgAlgos.CSPadArrPeakFinder]   N processed events = 3000  N selected = 81  Fraction of selected = 0.027
[info:ImgAlgos.CSPadArrPeakFinder]   N processed events = 4000  N selected = 95  Fraction of selected = 0.02375
[info:ImgAlgos.CSPadArrPeakFinder]   N processed events = 5000  N selected = 150  Fraction of selected = 0.03
[info:ImgAlgos.CSPadArrPeakFinder]   N processed events = 6000  N selected = 265  Fraction of selected =
0.0441667
[info:ImgAlgos.CSPadArrPeakFinder]   N processed events = 7000  N selected = 404  Fraction of selected =
0.0577143
[info:ImgAlgos.CSPadArrPeakFinder] ===== JOB SUMMARY =====
[info:TimeInterval::stopTime] Time to process 7945 events is 3747.48 sec, or 0.471678 sec/event
```

Selected events

ev-007713:

ev-008944:

Next example shows how to use ImgAlgos::CSPadArrPeakFinder in combination with python module pyimgalgos.ex_peaks_nda:

```
[psana]
#skip-events = 50
events      =  10
files = exp=cxi83714:run=136

modules = cspad_mod.CsPadCalib \
          ImgAlgos.CSPadArrPeakFinder \
          pyimgalgos.ex_peaks_nda
#         EventKeys

[cspad_mod.CsPadCalib]
inputKey      =
outputKey     = calibrated
doPedestals   = yes
doPixelStatus = yes
doCommonMode  = yes

[ImgAlgos.CSPadArrPeakFinder]
source            = DetInfo(CxiDs1.0:Cspad.0)
key               = calibrated
key_peaks_out     =
key_peaks_nda     = peaks_nda
hot_pix_mask_inp_file = ana-cxi83714/cspad-cxi83714-r0136-noise-mask-ini.dat
hot_pix_mask_out_file = ana-cxi83714/cspad-cxi83714-r0136-noise-mask-out.dat
frac_noisy_evts_file  = ana-cxi83714/cspad-cxi83714-r0136-noise-frac.dat
evt_file_out        = cxi83714/cspad-ev-
rmin                =     8
dr                  =     1
SoNThr_noise        =     3
SoNThr_signal       =     4
frac_noisy_imgs     =   0.9
peak_npix_min       =     3
peak_npix_max       =   500
peak_amp_tot_thr    =     0.
peak_SoN_thr        =     5.
event_npeak_min     =     5
event_npeak_max     =  1000
event_amp_tot_thr   =     0.
nevents_mask_update =     0
nevents_mask_accum  =    50
selection_mode      = SELECTION_ON
out_file_bits       =    15
print_bits          =  3681

[pyimgalgos.ex_peaks_nda]
source        = DetInfo(CxiDs1.0:Cspad.0)
key_in        = peaks_nda
print_bits    = 255
```

Example for module ImgAlgos::CSPadArrPeakAnalysis

See Module ImgAlgos::CSPadArrPeakAnalysis

Example of the psana configuration file:

```
modules = cspad_mod.CsPadCalib \
          ImgAlgos.CSPadMaskApply \
          ImgAlgos.CSPadArrPeakFinder \
          ImgAlgos.CSPadArrPeakAnalysis

# ...configuration parameters of other modules...

[ImgAlgos.CSPadArrPeakAnalysis]
source             = DetInfo(CxiDs1.0:Cspad.0)
key                = peaks
print_bits         = 7
fname_root         = file.root
```

After execution in psana the `file.root` containing histogram(s) and ntuple(s) will be produced. Then, auxiliary script in root, running by the commend

```
root -q -f proc.C
```

produces the plots with histograms:

-

Example for TimeStampFilter and XtcOutputModule

This example demonstrates how to run psana with the **time stamp filter** and **event writer** in xtc file.

Both modules are available in psana library and they need only to be described in the configuration file. For example, the configuration file `tstamp-filter-and-event-writer.cfg` may looks like:

```
[psana]
files         = /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/e158-r0021-s00-c00.xtc \
                /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/e158-r0021-s01-c00.xtc \
                /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/e158-r0021-s02-c00.xtc \
                /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/e158-r0021-s03-c00.xtc \
                /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/e158-r0021-s04-c00.xtc \
                /reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/e158-r0021-s05-c00.xtc

skip-events   = 10
events        = 100
modules       = ImgAlgos.TimeStampFilter  PSXtcOutput.XtcOutputModule

[PSXtcOutput.XtcOutputModule]
dirName       = ./test_out

[ImgAlgos.TimeStampFilter]
tsinterval    = 2012-02-02 18:17:00.409143728 / 2012-02-02 18:17:00.525853474
filterIsOn    = true
print_bits    = 11
```

Command to run psana is:

```
psana -c ./tstamp-filter-and-event-writer.cfg
```

For this configuration file psana will skip 10 events and loop over the next 100 events from the `/reg/d/psdm/<INSTRUMENT>/<experiment>/xtc/e158-r0021-s0*-c00.xtc` files and run consecutively `modules = ImgAlgos.TimeStampFilter PSXtcOutput.XtcOutputModule`. Parameters of these modules are described in the bottom part of the configuration file. Module `TimeStampFilter` passes events from the specified time interval and prints some useful information. Module `XtcOutputModule` will write passed events in the file with auto-generated name `./test_out/e158-r0021.xtcf`.

See also: Module ImgAlgos::TimeStampFilter and PSXtcOutput::PSXtcOutput

Example for module ImgAlgos::UsdUsbEncoderFilter

Example of configuration file:

```
[psana]
files = exp=amo75113:run=156
events = 10

modules = ImgAlgos.UsdUsbEncoderFilter

[ImgAlgos.UsdUsbEncoderFilter]
source  = DetInfo(:USDUSB)
mode    = 1
ifname  = tstamp-code-in.txt
ofname  = tstamp-code-out.txt
print_bits = 31
```

See also: Module ImgAlgos::UsdUsbEncoderFilter

Examples for Package pyimgalgos

Package **pyimgalgos** contains python modules which work with both frameworks **pyana** and **psana**.

Configuration file for **pyana** and/or **psana** should have relevant sections with parameters for `[pyana]` and/or `[psana]`. Alean section is ignored in each framework at run time. This is the only difference between two frameworks in the configuration file. All module descriptions are the same for two frameworks, as shown in examples below.

See description of modules in Package pyimgalgos.

Example of configuration file for CSPAD

File `py-xcs72913-r0049-cspad.cfg`:

```
# Run this script:
# psana -c py-xcs72913-r0049-cspad.cfg
# pyana -c py-xcs72913-r0049-cspad.cfg
#
# Useful commands:
# psana -m EventKeys -n 5 /reg/d/psdm/xcs/xcs72913/xtc/e265-r0049-*
# psana -m psana_examples.dump_cspad -n 5 exp=xcs72913:run=49
# pyana -m pyana_examples.dump_cspad -n 5 /reg/d/psdm/xcs/xcs72913/xtc/e265-r0049-*

[pyana]
files  = /reg/d/psdm/xcs/xcs72913/xtc/e265-r0049-s00-c00.xtc /reg/d/psdm/xcs/xcs72913/xtc/e265-r0049-s04-c00.
xtc /reg/d/psdm/xcs/xcs72913/xtc/e265-r0049-s05-c00.xtc
num-events = 5
#skip-events = 0
#num-cpu = 1
verbose = 1 ; logging output: 0-nothing?, 1+INFO, 2+DEBUG, ...
modules = pyimgalgos.tahometer pyimgalgos.cspad_arr_producer pyimgalgos.cspad_image_producer pyimgalgos.
image_save_in_file


[psana]
files = exp=xcs72913:run=49
events = 5
#skip-events = 0
modules = pyimgalgos.tahometer pyimgalgos.cspad_arr_producer pyimgalgos.cspad_image_producer pyimgalgos.
image_save_in_file
verbose = 1


[pyimgalgos.tahometer]
dn         = 10
print_bits = 255


[pyimgalgos.cspad_arr_producer]
#source    = -|Cspad-*
source     = XcsEndstation-0|Cspad-0
data_type  = double
#data_type  = float
#data_type  = unsigned
#data_type  = uint16
val_miss   = 0
key_out    = cspad_array
print_bits = 1


[pyimgalgos.cspad_image_producer]
calib_dir  = /reg/d/psdm/xcs/xcs72913/calib/CsPad::CalibV1/XcsEndstation.0:Cspad.0/
key_in     = cspad_array
key_out    = cspad_image
print_bits = 1

# Supported output file formats tiff, gif, png, eps, jpg, jpeg, txt, npy(default), npz
[pyimgalgos.image_save_in_file]
key_in     = cspad_image
ofname     = img-for-cspad.txt
print_bits = 255
```

To run this script use command

> *psana -c py-xcs72913-r0049-cspad.cfg*

or

> *pyana -c py-xcs72913-r0049-cspad.cfg*

Example of configuration file for CSPAD2x2

File `py-meca6113-r0028-cspad2x2.cfg`

```
# Run this script:
# psana -c py-meca6113-r0028-cspad2x2.cfg
# pyana -c py-meca6113-r0028-cspad2x2.cfg
#
# Useful commands:
# psana -m EventKeys -n 5 /reg/d/psdm/mec/meca6113/xtc/e332-r0028-s03-c00.xtc
# psana -m psana_examples.dump_cspad -n 5 exp=meca6113:run=28
# pyana -m pyana_examples.dump_cspad -n 5 /reg/d/psdm/mec/meca6113/xtc/e332-r0028-s03-c00.xtc

[pyana]
files  = /reg/d/psdm/mec/meca6113/xtc/e332-r0028-s03-c00.xtc
num-events = 5
#skip-events = 0
#num-cpu = 1
verbose = 0 ; logging output: 0-nothing?, 1+INFO, 2+DEBUG, ...
modules = pyimgalgos.tahometer pyimgalgos.cspad_arr_producer pyimgalgos.cspad_image_producer pyimgalgos.
image_save_in_file


[psana]
files = exp=meca6113:run=28
events = 5
#skip-events = 0
modules = pyimgalgos.tahometer pyimgalgos.cspad_arr_producer pyimgalgos.cspad_image_producer pyimgalgos.
image_save_in_file


[pyimgalgos.tahometer]
dn         = 10
print_bits = 255


[pyimgalgos.cspad_arr_producer]
#source    = -|Cspad-*
source     = MecTargetChamber-0|Cspad2x2-3
#data_type  = double
data_type  = float
#data_type  = unsigned
#data_type  = uint16
#data_type  = uint32
val_miss   = 0
key_out    = cspad2x2_array
print_bits = 255


[pyimgalgos.cspad_image_producer]
calib_dir  = /reg/d/psdm/mec/meca6113/calib/CsPad2x2::CalibV1/MecTargetChamber.0:Cspad2x2.3/
key_in     = cspad2x2_array
key_out    = cspad2x2_image
print_bits = 1

# Supported output file formats tiff, gif, png, eps, jpg, jpeg, txt, npy(default), npz
[pyimgalgos.image_save_in_file]
key_in     = cspad2x2_image
ofname     = img-for-cspad2x2.tiff
print_bits = 255
```

To run this script use command

> *psana -c py-meca6113-r0028-cspad2x2.cfg*

or

> *pyana -c py-meca6113-r0028-cspad2x2.cfg*

References

psana - Module Catalog