OPTIONS AND ARGUMENTS OF THE SAXS PROGRAMS

Options and Arguments of the SAXS Programs

Version: 2013-03-06

Introduction

The saxs programs have been designed for operations on sequences of 2d X-ray scattering data, e.g. to add or to multiply sequences of image arrays. They are also used for operations on single images. In general, there exist a separate saxs-program for each basic operation, called saxs_add (addtion), saxs_mul (multiplication), saxs_div (division), saxs_mac (multiplication with a factor and addition of a constant). There are also programs with special functionality, e.g. saxs_arc to transform a 2d cartesian scattering pattern into polar coordinates, saxs_row to project a selection of rows to a single row, saxs_col to project a selection of columns to a single column, etc. The list of programs can be found in the section SAXS Programs and Specific Options.
The ESRF data format (.edf) is used for reading and writing data files. This format allows the storage of metadata, image data and variance data in the same file. The header can be inspected by opening it with a text editor or the unix command less. But be careful, saving a so opened file afterwards corrupts usually its structure and makes it unusable.
It is generally possible to read BSL data files when they have been written in the same byte order as the machine on which the saxs-programs are running, if not change it with the options -bibo 1 or 2. There exist some specific programs to convert image data into edf data files, e.g. with the utility binary2saxs or the utility ascii2saxs.

Parameters can be passed in three ways to the saxs programs:

a) by prompts

b) by arguments

c) by options

The multiplication of the first three images of the input file "x.edf" with the factor 5.5 is used to demonstrate these possibilities. The output is written to "y.edf".

If a read-only input file terminates with .gz or .Z it is automatically uncompressed before reading. If a new output file (option -omod n) terminates with .gz or .Z it is compressed after closing.

Prompts

The program prompts for some important parameters, but not for all.

- `> saxs_mac`

  Usage: saxs_mac [options]

  <i1nam> <onam> <i1fst> <i1lst> <i1inc>

  <odum> <odim1> <odim2> <ofac> <ocon> <shft1> <shft2>

saxs_mac saxs_mac 3.12 24-May-1995, Peter Boesecke

Input sequence 1 [input.edf] : x.edf <return>
Output sequence [output.edf] : y.edf <return>
First image of input sequence 1 [1] : <return>
Last image of input sequence 1 [1] : 3 <return>
Increment of input sequence 1 [1] : <return>
Output dummy [0.000000e+00] : -1 <return>
Output dimension 1 [1024] : <return>
Output dimension 2 [1024] : <return>
<output image> = <input image> * Factor + Const
Multiplication factor [1.000000e+00] : 5.5 <return>
<output image> = <input image> * Factor + Const
Addition constant [0.000000e+00] : <return>
Input reference system is Image
output shift 1 [0.000000e+00] :
Input reference system is Image
output shift 2 [0.000000e+00] :
...
...

Arguments

The same parameters can be passed as arguments in the same order as the program would prompt for them. "=" stands for an empty argument. The option -p is used to switch off the prompt mode. With +p the program will prompt and will show the arguments as defaults in parenthesis []. This can be used for checks.

- `saxs_mac -p x.edf y.edf = 3 = -1 = = 5.5 = = =`

Options

Usually, the default of all arguments can be set with options. While arguments must be passed in a specific order options can be passed in any order before the first argument. Options are preceeded by "-" or "+" and can be repeated. The last occurence sets the value.

- `saxs_mac -p -i1nam x.edf -onam y.edf -i1lst 3 -odum -1 -i1fac 5.5`

Examples

Subtraction of two images:

- saxs_sub image1.edf image2.edf output.edf

  If image1.edf and image2.edf contain several images the first image in image2.edf is subtracted from all images in image1.edf and written to output.edf.

Divide image1.edf by image2.edf

- saxs_div.edf image2.edf output.edf

Add image1.edf and image2.edf

- saxs_add image1.edf image2.edf output.edf

Multiply image1.edf with image2.edf

- saxs_mul image1.edf image2.edf output.edf

Conversion to polar coordinates (azimuthal regrouping)

- saxs_arc image1.edf output.edf

Projection (averaging) of several column

- saxs_row image1.edf output.edf

Text file output

- saxs_curves image1.edf

The output will be written to image1.txt

More complex operations between images are possible, e.g. subtraction of series of background images from series of images. In these cases special options must be used.

Some Useful Commands

In the following examples all header values of the input images are passed to the output images (+pass). The user will not be prompted (-p) and existing output files will be overwritten (-omod n). The command will be automatically applied to a series of input files.

In the first example the intensity of the input image is normalized to the header value Intensity1 (-scal) and then divided by by a flat-field image (+flat -i3nam flat.msk). The intensity of each pixel is divided by its spherical angle (calculated from the header values Center_1, Center_2, Psize_1, Psize_2 and SampleDistance). The output image is binned by 4 in each direction (-obin 4 4) and finally multiplied by 1.8/2.75 (-ofac 1.8/2.75).

This operation is done for all input files stp3_0002.edf to stp3_0039.edf (-i1nam stp3_%%%.edf,2,39). The results are written to the files stp3_0002.nrm to stp3_0039.nrm (-onam stp3_%%%.nrm).

- saxs_normn -p -ofac 1.8/2.75 -omod n +pass +flat -i3nam flat.msk -scal \
  -i1nam .. stp3_%%%.edf,2,39 -onam stp3_%%%.nrm -obin 4 4

If the values of center, pixel size and sample distance are not correct in the input images they can be set with the options -i1cen <Center_1[pc]> <Center_2[pc]> -i1pix <Psize_1[m]> <Psize_2[m]> -i1wvl <WaveLength[m]> (pc = pixel coordinate). Attention, the length unit is always meter.

The following command performs an azimuthal regrouping of the input image. Axis 1 of the output image corresponds to the radius, axis 2 to the azimuthal angle. The default creates 360 sections of 1 degree that are written to a row of the output image. The center and pixel sizes must be correctly defined in the header, otherwise the options shown above (-i1cen and -i1pix) must be used to update them.

This operation is done for all input files stp3_0002.nrm to stp3_0039.nrm (-i1nam stp3_%%%.nrm,2,39). The results are written to the files stp3_0002.ang to stp3_0039.ang (-onam stp3_%%%.ang). In this case the input and output files are passed as arguments in the same way the program would prompt for them.

- `saxs_angle -omod n -p +pass -rsys normal \`
  `stp3_%%%.nrm,2,39 stp3_%%%.ang`

The following command averages over all rows of the input image, in this case the azimuthally regrouped scattering pattern where each line corresponds to a sector. Like in the previous cases the operation is applied to all images between stp3_0002.ang stp3_0039.ang and the file names are passed as arguments. Each output image contains only a single row, the azimuthal average.

- `saxs_row +pass -omod n -p stp3_%%%.ang,2,39 stp3_%%%.row`

The last command converts the azimuthal average to a text file with two columns (+swap) that are separated by a tabulator (ASCII TAB character). The first column contains the scattering vector q in 1/nm, the second column the intensities. The default for scattering vectors is s=2sin(Theta)/WaveLength (in 1/nm). The option -scf "2*pi" multiplies the scattering vector by 2pi: 2*pi*s = q (in 1/nm). The scattering vectors are calculated using the scattering angle and trigonometric functions (+waxs) and not an approximation for small angles. The table is preceded by a summary of header values, e.g. the data history. Column labels (-headl "lab1 lab2 ...") are written above the columns. The first column label is "#q*nm", the second column label is "I([WaveLength]m)", where [WaveLength] will be replaced by the header value of WaveLength. The hash (#) is only added to allow a plot with gnuplot.

- `saxs_ascii -scf "2*pi" -p +waxs +swap -hedl "#q*nm I([WaveLength]m)"\`
  `stp3_%%%.row,2,39`

The output file looks like

```
#{
#Image =  1
#Size =  756
...
#History-4 = saxs_row +pass -omod n -p stp3_%%%.ang,2,39 stp3_%%%.row
#
#
#
#}
#q*nm   I(1.75114e-10m)
0.0136098       -10
```

0.0408293 -10

0.0680488 -10

0.0952682 -10

0.122488 -10

0.149707 6.08817

0.176926 7.82687

...

Geometry

The geometry allows the description of 2-dimensional (rastered) data from detectors oriented in 3-dimensional space. Some of the concepts can be extended to N-dimensions.

Coordinate System and Orientation in Space

The observer is looking from the sample to the detector. The directions are numbered from 1 to 3. If the detector is perpendicular to the primary beam axis 1 points horizontally to the right, axis 2 points upwards and axis 3 is the cross product of axis 1 with axis 2. All three axes built a right handed orthogonal coordinate system. The position of the detector relative to the scattering center (goniometer center) is described by the length of the normal on detector that points to the scattering center. The pixel coordinate of this normal on the detector is the point of normal incidence, which for historical reasons is also called center. The rotation of the detector around the scattering volume is generally described by a 3d rotation matrix. The rotation matrix is described with three subsequent ccw rotations around axis 1 to 3 in the laboratory system: rotation 1 , rotation 2 and rotation 3.

Raster Orientation

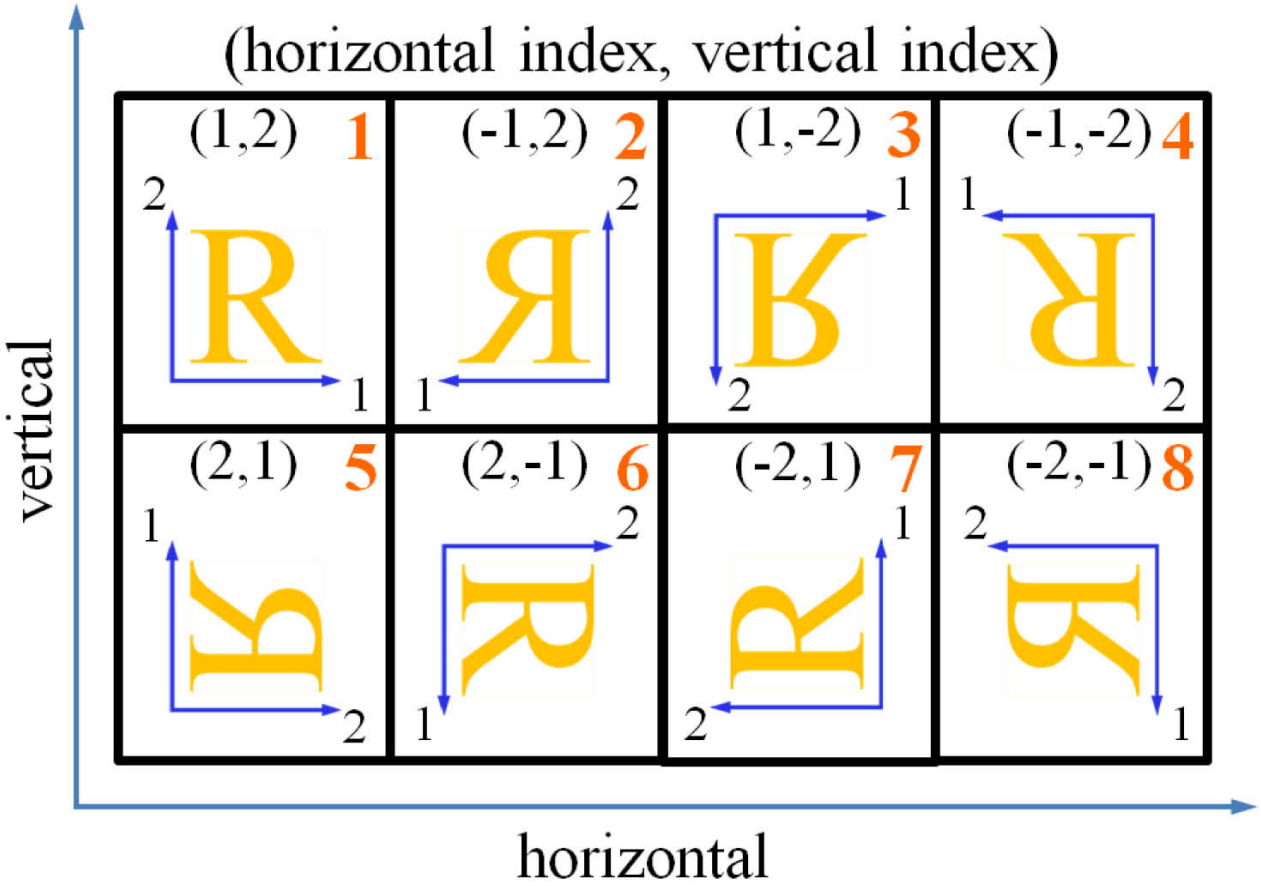

Fig. 1. Raster Orientations for N=2 (two-dimensional image). Raster orientation number 1 is used as reference. It is defined as the orientation where the geometrical origin of the image is at the lower left corner, where index 1 corresponds to the horizontal coordinate (x_1) and where index 2 corresponds to the vertical coordinate (x_2). All other orientations are derived from orientation 1 by inverting and transposing the coordinates. In raster orientation 5 index 1 corresponds to the vertical coordinate and index 2 to the horizontal coordinate.

The orientation of the image describes the relation between data indices and directions in real space. In orientation 1 index 1 corresponds to the horizontal axis and index 2 to the vertical axis. Internally, all images are converted to orientation 1. The conversion includes a reorientation of the image and a swapping of some header values, e.g. for dimension, pixel size and offset. The orientation of the input and output image can be chosen with the options -i1ori O and oori O, where O is the orientation number shown in the upper right corner of Fig. 1.

Pixel Coordinates

Coordinates are used to specify fractional positions in arrays. To avoid unwanted shifts, e.g. of the center, it is important to use a single coordinate definition. The choice is arbitrary, but to set the lower edge (p0) to zero facilitates the use of region of interests. In this way the position along each axis of an image array is described in the following way (see Fig. 1).

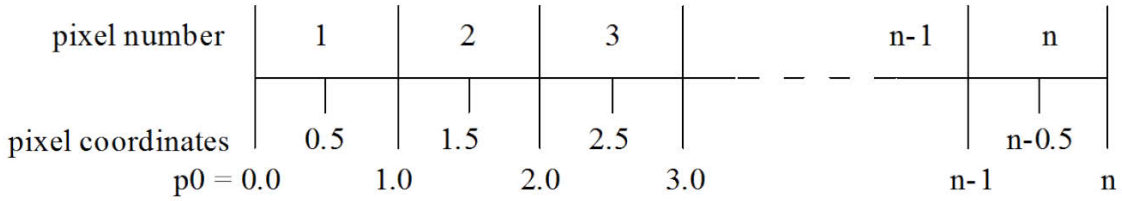

Fig. 2. Pixel coordinates (p0=0.0: pixel coordinate of lower edge, n: pixel coordinate of upper edge).

The width of each pixel is defined as one. With this definition pixel number 1 covers the pixel coordinate range between 0.0 and 1.0, pixel number 2 from 1.0 to 2.0 and the last pixel the pixel coordinate range between n-1 and n. The centers of pixels are 0.5, 1.5, etc.

Reference System Coordinates

A reference system describes a specific affine transformation of a pixel coordinates p to a reference system coordinate c (1). The scale factor a and the offset b are functions of the header parameters Offset, Center, BSize, PSize, SampleDistance and Wavelength.

(1) $c = a\ p + b$

This concept is used to change pixel coordinates to coordinates in meter or to use a specific center etc. without changing the header parameters. During calculation the same reference system is used for all images, e.g. to align them all with respect to their centers. The default reference system is image, e.g. coordinate offsets are taken into account.

Currently, eight different reference systems are defined: array, image, center, region, real, normal, tangens, saxs. Each reference system uses a specific affine transformation as a function of header parameters (shown in bold)..

array: ARRAY coordinate = pixel coordinate p

image: IMAGE coordinate = ARRAY coordinate + **Offset**

center: CENTER coordinate = IMAGE coordinate - **Center**

region: REGION coordinate = IMAGE scoordinate * **BSize**

real: REAL coordinate = IMAGE coordinate * **PSize**

normal: NORMAL coordinate = CENTER coordinate * **PSize**

tangens: TANGENS coordinate = NORMAL coordinate / **SampleDistance**

saxs: SAXS coordinate = TANGENS coordinate / (**Wavelength** * 1e-9)

A reference system is used to define the relative geometrical alignment between images. It can be chosen with the option -rsys <refsys> (see chapter Option Values).

Parameters and Keywords

All saxs-programs use a common set of parameters to describe the geometry. These parameters are written as key-value pairs into the data file header. A parameter name followed by an underscore and a number refers to a specific array coordinate, i.e. Dim_1 is the length of the array in the first coordinate, Dim_2 is the length of the array in the second coordinate. The dimension (rank) N of the array is given by the key Dim_N with the highest N, i.e. Dim_2 for 2d arrays. Generally, SI units are used and not specified.

Keyword Describing the Orientation of the Coordinate System

- Raster Orientation: (integers 1..8)

Keywords Describing Region of Interests

- Dim_1, Dim_2: (integer) lengths of the image data array in direction 1 and 2
- Offset_1, Offset_2: (float) [array coordinate] offsets of coordinates 1 and 2
- BSize_1, BSize_2: (float) binning sizes, e.g. 2, 2 after 2x2 binning of the raw detector image

Keywords Describing the Scattering Geometry

- Center_1, Center_2: (float) [image coordinate] position of the point of normal incidence (PoNI)
- SampleDistance: (float) [m] distance between point of normal incidence and sample
- PSize_1, PSize_2: (float) [m] pixel sizes
- WaveLength: (float) [m] wavelength

Keywords Describing the Scattering Image of Rotated Detectors

- DetectorRotation_1: (float) [rad] first detector rotation (ccw around lab axis 1)
- DetectorRotation_2: (float) [rad] second detector rotation (ccw around lab axis 2)
- DetectorRotation_3: (float) [rad] third detector rotation (ccw around lab axis 3)

Axis Types

Each axis can have a different type. Three types of axes are defined: Distance (default), Angle and Numerator. Distances and angles are measurements in real or reciprocal space. Numerator is any numbering, e.g. in a composed image where each line refers to a different frame. If numbers are requested 1 means Distance, 2 Angle and 3 Numerator.

Cartesian Coordinates (2D Detector Pattern, Distance/Distance)

This is the most common case. All axes coordinates are proportional to distances (AxisType Distance).

Polar Coordinates (Azimuthally Regrouped Pattern, Distance/Angle)

In this case axis 1 is the radius (AxisType Distance) and axis 2 is the angle (AxisType Angle) or arc (AxisType Distance).

Line by Line Compositions (Distance/Numerator)

In this case each line of the 2D pattern corresponds to a 1D pattern, e.g. an azimuthal average of a 2D detector pattern and axis 2 is the frame number. In this case axis 1 is radius (AxisType Distance) and axis 2 a numerator.

Projection Types

Two projection types are defined: Saxs (default) and Waxs. If numbers are requested 1 means Saxs and 2 Waxs. Projection type Saxs means that distances (axis type distance) are measured in real space, projection type Waxs means that distances are measured in reciprocal space. In the limit of small angle scattering ($2\sin(\vartheta) \approx 2\vartheta \approx \tan(2\vartheta)$) both projection types converge. In projection type Waxs the projection plane is always perpendicular to the incident beam. Detector rotations are not used (see program saxs_waxs).

List of Options

Each option in the following list is followed by a parenthesis that shows the number of parameters and the parameter type:

option [<num of parameters>,<typ>]

The saxs programs are called in the following way:

saxs_xxx [option] {[option]} [argument] {[argument]}

Options are preceeded by '-' or '+'. Options are read from left to right

and are processed before the arguments are read. Options can be repeated.

The last option sets the value.

Example:

saxs_normn +flat -i3nam "flat_field" -trm 0.78 -ofac 1/0.0003 \

image.edf = = image.nrm

Each option can be followed by parameters. The number of parameters

and the type of the parameter are shown in brackets. The following parameter

types are used:

Option Values

+: boolean value , e.g. +option and -option (no parameter)
d: integer value, e.g. -ofst 3
f: float value, e.g. -ofac 1/0.0003
M: file opening mode (n+fp, n+ip, n, o+fp, o+ip, o, a+fp, a+ip, a)

- n, n+fp : new (create new file and empty any old file, allow to overwrite existing images if header size and image size fit into the existing block structure)
- n+ip : new + image protection (create new file, do not allow to overwrite existing images)
- o, o+fp : old (use old file, error if it does not exist, allow to overwrite existing images if header size and image size fit into the existing block structure)
- o+ip : old + image protection (use old file, do not allow to overwrite existing images)
- a, a+fp : any (if the file does not exists create a new file, otherwise open it as an old file, allow to overwrite existing images if header size and image size fit into the existing block structure)
- a+ip : any + image protection (like any but do not allow to overwrite existing images)

P: projection type value
(Saxs, Waxs), e.g. -opro Waxs

R: reference system value
(array, image, center, real, region, normal, saxs), e.g. -rsys saxs

- array: ARRAY coordinate p = pixel coordinate
- image: IMAGE coordinate i = ARRAY coordinate + offset
- center CENTER coordinate i = IMAGE coordinate - center
- region REGION coordinate i = IMAGE coordinate * bin size

- real: REAL coordinate r = IMAGE coordinate * pixel size
- normal NORMAL coordinate i = CENTER coordinate * pixel size
- tangens TANGENS coordinate i = NORMAL coordinate / (sample distance)
- saxs: SAXS coordinate s = (NORMAL coordinate) / (sample distance) * WaveLength0/wavelength (WaveLength0=1e-9 m)

s: string value, e.g. -otit "flatfield image"

Arithmetic Expressions as Values

Integer and float values can be entered as arithmetic expressions. Common units are defined in SI-units, e.g. keV is converted to Joule. This allows an easy conversion of values. In the following way the wavelength can be calculated from the photon energy and entered into the program `-i1wvl "h*c/12.5_keV"`. The quotations marks must be set to avoid an interpretation by the command shell.

Additional functions and constants are defined. To get a full list of all defined constants and units type "f`calc debug=2 a`" on the command line. Units can be appended with an underscore to a number, e.g. 1_nm for 1e-9_m. The multiplication with a unit is only the multiplication with a conversion constant to a base unit, e.g. nm=1e-9, m=1.

Arithmetic and Logical Operators

Comparisons, logical operators and conditional expressions of the form a?b:c are possible e.g.

```
-i1cen '[SampleDistance]<4_m?200:300' '[SampleDistance]<4_m?200:300'
```

If [SampleDistance] is smaller than 4_m the value is

```
-i1cen 200 200
```

otherwise

```
-i1cen 300 300
```

Other examples:

```
fcalc '(4_m>399.9_cm)&&(6_m!=10_m)'
1

fcalc '1||2&&3!=(4==5)<(((6<=7)>8)>=9)'
1
```

The operators have the following hierarchy (strongly binding => less binding):

high

parentheses:	(,) etc.
unit multiplicator:	_
unary operator:	!
binary arithmetic operators:	*, /, %
binary arithmetic operators:	+,-
comparison:	<, <=, >, >=
equality:	!=, ==

logic product: &&
logic sum: ||
condition: a?b:c
low

Constants

Physical and mathematical constants are taken from:

[86] Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720
X-Ray data booklet, second printing, with corrections, April 1986

Constants that are not contained in this edition are taken from
[91] Kuchling, Taschenbuch der Physik, Verlag Harri Deutsch 1991
There are currently some inconsistencies in some definitions, e.g. amu, ga and p.

```
gamma = 0.577215664901532861;
e     = 2.71828182845905 /* Euler number */
k     = 1.380662e-23;    /* Boltzmann constant (J/K) */
me    = 9.109534e-31;    /* electron rest mass (kg) */
mp    = 1.6726485e-27;   /* proton rest mass (kg) */
NA    = 6.022045e23;     /* Avogadro number (1/mol) */
pi    = 3.14159265358979323846;
re    = 2.8179380e-15;   /* classical electron radius (m) */
c     = 2.99792458e8;    /* velocity of light (m/s) */
ec    = 1.6021892e-19;   /* electron charge (C) */
h     = 6.626176e-34;    /* Planck  constant (J*s) */
```

Units

```
b       = 1e-28;        /* barn (1/m^2) */
amu   = 1.6605655e-27; /* atomic mass unit (kg) */
keV   = ec*1e3;        /* keV (J) */
eV    = ec;            /* eV (J) */
km    = 1e3;           /* km (m) */
m     = 1.0;           /* (m) */
cm    = 0.01;          /* cm (m) */
mm    = 1e-3;          /* mm (m) */
um    = 1e-6;          /* um (m) */
nm    = 1e-9;          /* nm (m) */
```

Arithmetic Functions

Float Functions

```
rad(x)    = pi/180.0 * x; /* transformation deg to rad */
deg(x)    = 180.0/pi * x; /* transformation rad to deg */
pi(x)     = pi;
sin(x)
cos(x)
tan(x)
```

asin(x)
acos(x)
atan(x)
atan2(x,y)
sinh(x)
cosh(x)
tanh(x)
floor(x)
ceil(x)
abs(x)
exp(x)
log(x)
log10(x)
pow(x)
sqrt(x)
round(x) = floor(x + 0.5); /* rounding */

Long integer functions

true = 1l;
false = 0l;
yes = 1l;
no = 0l;
(long int) x : transformation of x to long integer;
(round) x : rounding and transformation of x to long integer;

General Options Available in All SAXS Programs

An option can be followed one or more parameters. All options are described in the following way:

<option name> [<number_of_parameters>, <parameter type>]

aa [1, +]
interpolation method for anti-aliasing: (default antialiasing: +aa).

This interpolation method for small regions inside a pixel has been implemented to avoid that a wedge-like intensity distribution is transformed into a staircase-like intensity distributions during unbinning. The value of an output pixel that is calculated from an area smaller than an input pixel is averaged over a full pixel. The value and variances are scaled to match the size of the output pixel. This avoids sharp intensity jumps between regions of neighbouring pixels.
test [0, +]
switches a test modus on (+test) and off (-test)

add [1, d]
sets the number of successive images that should be added (see Add Successive Images)
Example: "saxs_mac +add 2 in%%.edf,1,10 out%%.edf" adds 2 subsequent images.
(default: 1)

bibo [1, d]
selects globally the input byte order of bsl files. All bsl files that are read by a single routine must have the same byte order, allowed values: 1 for low byte first (Intel, AMD), 2 for high byte first (Motorola, SUN) (default: 0 uses internal byte order of the machine)

cmpr [1, d]

set compression mode of binary blocks in output images, possible values are None (1), Gzip (2) , Z (3), e.g. saxs_mac -cmpr Z input.edf output.edf

dvo [1, d]
set datavalueoffset of output images. A data value offset shifts the data values in such a way that value(true)=value(stored)+dvo (attention, not generally supported). Only integer values are accepted. This allows negative values in unsigned data types, necessary for noisy values around zero.

gblk [1, +]
defines, whether the edf output file will be preceeded by a general block (+gblk) or not (-gblk) (default: -gblk) (see description of new edf format)

h [0, +]
displays long (+h) and short (-h) help

mhs [1, d]
sets the minimum header size. Headers will be at least mhs bytes long.

mlw [1, d]
sets the maximum line width in the output file. Lines longer than mlw will be terminated with a backslash and continued in the next line, default: not set.

p [0, +]
switches prompt mode on (+p) and off (-p)

pass [1, +]
pass all header values of first input image to output image

pmin [1, f]
sets the minimum ratio that a binned must overlap so that it is not set to dummy

rsys [1, R]
defines the reference system (-rsys array | image | real | saxs). The default value is image.

shft [2, f]
do not use, undefined behaviour

type [1, D]
output data type, currently, only the types marked with an asterisk are implemented
Conversion of file data types to internal data types

UnsignedByte (1 bytes)	unsigned char (1 bytes)
SignedByte (1 bytes)	char (1 bytes)
UnsignedShort (2 bytes) (*)	unsigned short (2 bytes)
SignedShort (2 bytes)	short (2 bytes)
UnsignedInteger (4 bytes)	unsigned long (4 bytes)
SignedInteger (4 bytes)	long (4 bytes)
FloatValue (4 bytes) (*)	float (4 bytes)
DoubleValue (8 bytes) (*)	double (8 bytes)

Example: "-type UnsignedShort"

usys [1, R]
defines the user reference system (-usys array | image | real | saxs). The default value is usys.

var [1, +]

create variance arrays (default: create variance arrays automatically when first input image has variances, otherwise not) (see also Error Propagation)

vw [1, +]

variance weighting: during interpolation, pixels are weighted inversely to their variance (default normal weighting -vw). Attention, use with care, this shifts the expectation value systematically to values with less variance.

+vw (variance weighting on)
-vw (variance weighting off) (default)

Inverse variance weighting can be used to sum/average pixels in patterns with pixels that have been inter/extrapolated from only very small portions of input pixels. When these values are included into sum they spoil the variance of the sum.

Inverse variance weighting is only implemented in the interpolation routines. It requires a variance array, otherwise the interpolation is done in the normal way. Inverse variance weighting is currently not used for binning, i.e. the options -iNbin/-obin perform always a normal binning with equal weight for each pixel.

Example

Inverse variance weighting can be used to calculate the average of an azimuthally regrouped image with saxs_row, e.g.

```
saxs_row +vw image.ang image.row
```

General Options for Output and Input Files

The following options can be used for output and for input files. An option for the output file starts with o, an option for the <n>-th input file starts with i<n>, e.g. -odum -1 -i1dum -1 -i2dum -1 etc. The number of input and output files can be viewed with the extended help option +h (block should be read as file):

```
...
Number of image blocks          :  3
Maximum number <n> of input blocks:  2
...
```

The file contains 1 output file (-onam <output file name>) and 2 input files (-i1nam <name of input file 1>, -i2nam <name of input file 2>).

Some conversion programs do not have standard input and output files, e.g. gas2saxs. In this case the number of image files ("blocks") is 0.

Image Numbers and Memory Numbers

Image numbers offer the possibility to store a sequence of (small) images in a single file. All images are numbered and can be stored in different memories. Image numbers and memory numbers are integer values. Positive memory values stand for image data, negative memory values

for error data. Memory zero must not be used. The default input memory number is 1, the default output memory number is the input memory number.

How image numbers can be chosen is described in the section Loop Control. Memory numbers can be chosen with the options -i<n>mem and -omem.

omem (i<n>mem) [1, d]
memory number of images in output (input) file, e.g. -omem -1.

File Access

omod (i<n>mod) [1, M]
output (input) file opening mode, e.g. -i1mod o -omod n+ip: open an existing file for reading and create a new output file, protect the images of the output file

onam (i<n>nam) [1, s]
output (input) file name, e.g. -i1nam "input.edf" -onam "output.edf"

The file "/dev/null" is used as dummy. Used as input, a saxs_xxx programs runs as if it would read from a file. All parameters can be defined explicitly by options, e.g. saxs_mac -i1dim 512 512 -i1con 5 /dev/null output.edf will create an image with 512 512 pixels and write it to output.edf. Each pixel has the value 5. Output to "/dev/null" suppresses the output to a real file.

Loop Control

All saxs_xxx programs read images sequentially from several input sequences (i1, i2, ...) and write the resulting images sequentially to a single output sequence (o). Per default the output images are described in the same way as the images of the first input sequence, e.g. title, pixel size etc. are copied from the input images of the first sequence. Only a single output image sequence is created: (Output[i01], Output[i02], ...). The output image is a function of the input images: Output[i0] = Function(image1[i1], image2[i2], image3[i3], ..., imageN[iN]). Per default, the start number (ofst), the increment (oinc) and the last number (olst) of the output image sequence are the same as start number (i1fst), increment (i1inc) and last number (i1lst) of the first input sequence. Thus, normally the output image number is equal to the number of the image in the FIRST input file.

i1fst, i1inc, i1lst -> maxloop1 = max(1,(i1lst-i1fst+1)/i1inc)

...

iNfst, iNinc, iNlst -> maxloopN = max(1,(iNlst-iNfst+1)/iNinc)

ofst, oinc, olst -> maxloop0 = max(1,(i0lst-i0fst+1)/i0inc)

maxloop = Min(loop1, .. , loopN, loop0)

If the increment of the output image is 0, maxloop0 is set to infinity and it is only written once at the end of the other loops.

If the increment of an input sequence J is zero the loop is stopped after the first loop (maxloopJ is set to 1).

The loop is stopped when the first incrementation loop has finished. Also negative increments are possible.

ImageLoop(...) {
for (O=OFirst,I1=I1First,I2=I2First,...;

```
    O<OLast,I1<I1Last,I2<I2Last,...;O+=OInc,I1+=I1Inc,I2+=I2Inc) {
  Read(I1,I2,...);
  Function((CmdBlk * ) pcb, (ImgHeadBlk *) ihb, (int *) pstatus );
  Write(O); }
 }
```

Missing images are skipped. All start images must exist.

ofst (i<n>fst) [1, d]
first image number of output (input) file, e.g. -i1fst 10 -ofst 10

olst (i<n>lst) [1, d]
last image number of output (input) file, e.g. -i1lst 20 -olst 20

oinc (i<n>inc) [1, d]
increment of image number of output (input) file, e.g. -i1inc 2 -oinc 2, if oinc (i<n>inc) is 0 the ouput image is only written once at the end of the loop (the input image is only read once)

Automatic Incrementation of File Numbers in a Loop

All saxs programs >= V4.0 allow incrementation of file numbers. The command

- `saxs_mac input%%%.edf,1,10,1 output%%%.edf`

processes the files input001.edf, input002.edf, input003.edf, ... input010.edf and writes the results to the files output001.edf, output002.edf, output003.edf, ... output010.edf.

Numbers in a file name are marked with place holders (percent signs %%%%%). The range of file numbers must be indicated by parameters after the file name, e.g.

input%%%.edf,<first>,<last>,<increment>

The parameters are separated by commas. Numerical expressions are possible, e.g.

input%%%.edf,1,1+20,2

Omitted parameters are replaced by default values. If possible, they are copied from a preceding input parameter, e.g.

saxs_mac input%%%.edf,1,20,1 output%%%.edf,,,2

The output file numbers will start with 1 (copied), will be incremented by 2 and will stop after 20 (copied):

output001.edf, ouput003.edf, ... output019.edf

Conversion between Multiple and Single Files

Multiple files with a single data section (images) can be written into a single file with multiple data sections by writing the output file name without placeholders for numbers, e.g.

- `saxs_mac input%%%.edf,1,20 output.edf`

will write input001.edf .. input020.edf to output.edf (image 1 to image 20). If the input data files contain itself several images, only the first image in an input file or the one specified with -i1fst is written into the output file.

In the same way a single file can be split into multiple files by giving placeholders in the output filename, e.g.

- saxs_mac input.edf output%%%.edf

will write all images of input.edf to numbered output files that contain only single images (default: image number 1).

Add Successive Images

Successive output images can be added before they are written. The option -add <n> will add n successive images (n>=1), e.g.

- saxs_mac -add 3 input%%%.edf,1,20 output%%%.edf

will add input001.edf to input003.edf and write the result to output001.edf,

will add input004.edf to input006.edf and write the result to output004.edf,

...

will add input016.edf .. input018.edf and write the result to output016.edf

Attention, the sum of input019.edf and input020.edf will be written to output020.edf. It will be the sum of only two images. Not all saxs-programs do really add the images, some may only output the last image. A test is necessary before using this option.

Transformation of Input and Output Images

After reading and before writing the pixel values of all images can be modified. The transformations are done in the following order after reading the input data and before writing to the output file. If the option is omitted nothing is done.

clip (-omin, -omax),

gnaw (-ogna),

rebin (-obin),

orientation (-oori)

transform (-ocon, -ofac))

Attention, each step can create new dummy pixels, e.g. -odum 10 -ofac 0 -ocon 10 would replace all output pixels with the value 10 which is the dummy value.

omin (i<n>min) [1, f]
: minimum value in output (input) image, e.g. -omin 0.01, lower values are replaced with dummies

omax (i<n>max) [1, f]
: maximum value in output (input) image, e.g. -omax 10000, larger values are replaced with dummies

ocon (i<n>con) [1, f]
: constant is added to output (input) image value before saving, e.g. -ocon 1000 (-i1con 1000), adds 1000 to each non-dummy pixel of an image before saving (after reading). Multiplication and addition are done at the same time according to: value' = value * factor + constant).

ofac (i<n>fac) [1, f]

factor is multiplied to output (input) image value before saving, e.g. -ofac 0.37 (-i1fac), multiplies each non-dummy pixel of an image with 0.37 before it is saved (after it was read). Multiplication and addition are done at the same time according to: value' = value * factor + constant).

obin (i<n>bin) [2, d]
rebinning factor of the output (input) images, e.g. -obin 2 2 rebins the output image by a factor 2. The image dimension is automatically adapted.

oori (i<n>ori) [1, d]
input: input image is transformed from orientation i<n>ori to
output: output image is transformed from orientation 1 to orientation oori

The value range of ori is from 1 to 8 (see Fig. 1)

ogna (i<n>gna) [2, d]
rounding of the dummy area of the output (input) images, e.g. -ogna 2 4 enlarges each dummy pixel by 2 pixels horizontally and by 4 pixels vertically.

oave (i<n>ave) [0, +]
This feature is only available for program versions 2.50 and higher. It switches between averaging and addition of rebinned or resized pixels. The values are calculated from partial pixel areas. The averages are done by dividing the sum by the sum of the partials of all contributing pixels.

- +iNave: averaging of rebinned and resized pixels (default)
- -iNave: addition of rebinned and resized pixels

Image Dimension

odim (i<n>dim) [2, d]
dimension of each output (input) image, e.g. -odim 720 1000 sets the output dimension horizontally to 720 and vertically to 1000. Unused pixels are filled with dummy. This option overwrites the header information of an input image.

In case of an input image this dimension is used instead of the value coming with the image.

Dummy Value

odum (i<n>dum) [1, f]
dummy value of each output (input) image, e.g. -odum -1 sets the output dummy value to -1. The value 0 is never a dummy. A value is a dummy if (abs(value - dummy) <= ddummy) (with ddummy > 0.1). Dummy values are ignored in calculations and are replaced by the output dummy value. This option overwrites the header information of an input image.

In case of an input image this dummy value is used instead of the value coming with the image.

oddum (i<n>ddum) [1, f]
ddummy value, radius around a dummy value, in which the pixel values are ignored. This value is automatically set and must be always larger than 0.1. There should be no need to use this option. See odum. This option overwrites the header information of an input image.

In case of an input image this ddummy value is used instead of the value coming with the image.

Additional Information

otit (i<n>tit) [1, s]
output (input) image title, e.g. -otit "black sample"

In case of an input image this title string is used instead of the title coming with the image.

otim (i<n>tim)
(*reserved for time string*)

Reference System Parameters

The reference system for calculation is globally chosen with rsys. The reference systems are described in a separate document.

ooff (i<n>off) [2, F]
offset of the output (input) image (in pixel)
This option overwrites the header information of an input image.

In case of an input image this offset is used instead of the offset given in the header..

ocen (i<n>cen) [2, F]
center of the output (input) image (in pixel). This option overwrites the header information of an input image.

In case of an input image this center is used instead of the center given in the header..

obis (i<n>bis) [2, F]
binning size of the output (input) image (multiples of pixel sizes). This option overwrites the header information of an input image.

In case of an input image this binning size is used instead of the binning size given in the header.

opix (i<n>pix) [2, F]
pixel size of the output (input) image (in meters). This option overwrites the header information of an input image.

In case of an input image this pixel size is used instead of the pixel size given in the header..

odis (i<n>dis) [1, F]
sample distance in the output (input) image (in meters). This option overwrites the header information of an input image.

In case of an input image this distance is used instead of the distance given in the header.

owvl (i<n>wvl) [1, F]
wavelength in the output (input) image (in meters). This option overwrites the header information of an input image.

In case of an input image this wavelength is used instead of the wavelength given in the header..

orot(i<n>rot) [3, F]
detector rotation angles in labs space around the sample with constant sample distance, first rotation around axis1 in lab space, then around axis 2 in lab space and finally around axis 3 in lab space (unit: radian). This option overwrites the header information of an input image.

In case of an input image these rotations are used instead of the rotations given in the header..

opro (i<n>pro) [3, P]

projection type of the output (input) image, values are "Saxs" or"Waxs".

In case of an input image this value is used instead of the projection value given in the header..

Replacement of Parameter Values with Header Values

Replacement of Parameter Values with Header Values (2003-05-25)

The parameters of some options can contain keys between square brackets. The keys and the square brackets are replaced by the header values of the keys. If a key cannot be replaced the parameter string is not changed and will probably cause an error when the parameter is a number. Only input header values can be used. The keys are searched in the corresponding input file. If a program has several input files the input file number (block number) can be explicitly written after the key separated by a comma. For example, to divide all values of input file 2 by the header value of Psize_1 in input file 2 one can write: -i2fac "1/[PSize_1]". Header values after output file options, e.g. -ofac "1/[PSize_1]", are replaced by header values of input file 1. A comma separated number after a key specifies the input file number in which the key should be searched, e.g. -i2fac "1/[PSize_1,1]" will search PSize_1 in input file 1.

Parameters that allow replacement by header values are marked in the help list (+h) by D and F. Usually all string parameters (s) allow replacement of header values. Exceptions are filenames. The following list shows parameters in which header values are replaced:

-shft, -i1/-omin, -i1/-omax, -i1/-ocon, -i1/-ofac, -i1/-obin,

-i1/-ogna, -i1/-otit, -i1/-otime, -i1/-odim, -i1/-odum, -i1/-oddum,

-i1/-ooff, -i1/-obis, -i1/-opix, --i1/-ocen, i1/-odis, -i1/-owvl, -i1/-orot

Syntax

A key value in an option is indicated with the name of the key enclosed in two brackets, e.g. [Dim_1] for the value of the header key Dim_1. The full syntax is:

<header value> := '[' <key> [',' <blkno> [',' <npar>]] ']' Where <blkno> is the block number, e.g. 1 for first input file (i1nam), 2 for second intput file (i2nam) etc. <npar> is the sequential numberof the parameter in the value string, e.g. for string options:

[Title,1] == parn1 parn2 parn3

[Title,1,0] == parn1 parn2 parn3

[Title,1,1] == parn1

[Title,1,2] == parn2

[Title,1,3] == parn3

The default for npar in number options is 1 (1st parameter only), for string options 0 (all parameters).

Currently, only the default delimiters (white-spaces) can be used. All parameters themselves can be defined as header values.

Special keys that return control parameters:

Special keys are case sensitive and are read before they are searched in the header. To read a header key with the same name change the case of the characters because header keys are case insensitive.

[FNAME] : file name

[FNUM] : file number

[MNUM] : memory number

[INUM] : image number

[LNUM] : loop number

Special keys are also available for the output block. Header keys and special keys use the same syntax in options, e.g. -otit "[Title] [FNAME,1]" appends the name of the input file to the title.

History Lines

Each file contains history lines that logs the calculation steps. The history keys are, starting with the oldest line, History-1, History-2 etc. The macros h1, h2, ... h9 extract History-1 to History-9. For higher levels history.mac <depth> <filename> can be used

Isotime Utilities

isotime2epoch

conversion of isotime string to epoch (seconds since 1970-01-01T00:00:00) (without leap seconds)

epoch2isotime

conversion of epoch to isotime (times before 1970-01-01 have negative epoch). Due to the 32-bit long integer format the time range is limited between
epoch=-pow(2,31) -> 1901-12-13T20:45:52.000000+0000 to
epoch=+pow(2,31)-1 -> 2038-01-19T03:14:07.000000+0000

The programs read from the standard input and write to the standard output. The can be used with the unix pipe command.

Error Propagation Options

General

var [0, +]

+var/-var : create/do not create variance arrays
(default: create only, if i1 image comes with variance array)

Input/Output Data:

oval expression(_oval), i<N>val expression(_i<N>val)
oerr expression(_oval,_oerr), i<N>err expression(_i<N>val,_i<N>err)

If one of these options is used error propagation is switched on. The option +val is not needed.

Calculation

The variances are propagated using the laws of Error Propagation. Cross correlations between pixels are ignored.

Contents

The variance array contains the variance of each pixel of the image array. The variance array has the same dimensions and orientation as the image array. The pixel [i,j] of the variance array contains the variance of pixel [i,j] of the image array. The variance header contains only keywords that describe an array. (Dim_1, Dim_2, DataType, ByteOrder, DataRasterConfiguration). The variance array does not use any additional keyword. If there are any, theyare ignored. The variance values are calculated after the calculation of the image values. Negative values in the variance array describe undefined variance values. The default value of a variance array is 0 (=exact).

Input and output variance arrays are only created if the first input image

If the option +var is set (default: not set) default variance arrays with variance zero are created for all input arrays. If the option -var is given no variance arrays are read/created and no error propagation calculation is done.

Storage

The variance array are stored in a second edf data block with the data block ID

EDF_DataBlockID = <image>.Image.Error[.<mem>].

The suffix .<mem> is omitted for memory=1.

For convenience, the data block of the variance array is written after the data block of the image and the header of the variance array of memory 1 contains additionally the keyword Image.Error = <image>, where <image> is the image number of the EDF_DataBlockID. The image header contains in this case the keyword Image = <image>.

Example

To define initial variances for the input array sphere.edf the following option must be given:

- `saxs_mac -i1err _i1val sphere.edf sphere.err`

It assumes that each pixel of sphere.edf contains the number of detected photons and that the errors follow Poisson statistics (var=σ^2=N). The option -i1err _i1val sets the variance to the pixel value (_i1val) of sphere.edf, _i1val is a variable. The following variables are defined:

_i1val: input pixel value

_i1err: input variance value

_oval: output pixel value

_oerr: output variance value

All variables start with an underscore.

Additional errors, e.g. var=1.5, can be added with.

- `saxs_mac -i1err _i1val+1.5 sphere.edf sphere.err`

To avoid negative variance values, the following command can be given:

- saxs_mac -i1err "max(0,_i1val)+1.5" sphere.edf sphere.err

SAXS Programs and Specific Options

ascii2saxs

Conversion of an ASCII table to an edf file

Arguments

ascii2saxs [-h] [options] <ascii file> {<ascii file>}

Options

verbose=0|1|2|... verbose
bskp=<number of bytes to skip> before start of the ascii data
lskp=<number of lines to skip> in the ascii data
cskp=<number of characters to skip> after line skips
skipcol=<number of columns to skip>
skiprow=<number of rows to skip>
dim=<dim1>x<dim2>: array dimensions
The dimensions can be specified with
dim=1x2 to set dim1 and dim2,
dim=x2 to set only dim2,
dim=1x to set only dim1.
The program tries to guess a dimension that is not set it.
ori=<rasterorientation>: raster orientation
dum=<dummy>: output dummy value
commentset=<commentset>: comment character set
delimiter=<delimiterset>: delimiter character set
+h : this help
pipe=0|1: read file names from stdin (stop with <ctrl-d>)
ext=<extension>: output file extension (default .edf)

binary2saxs

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Version

V4.02 2002-04-06

Arguments

binary2saxs [options] <fnam> <onam> <i1dim1> <i1dim2> <fskp> <ftyp> <fend> <fdvo> <fras> <odum>

Options

fskp [1, d]
number of bytes to skip

fnam [1, s]

binary file name

ftyp [1, s]
binary type : type of a single element

- "FloatValue" (old "float")
- "DoubleValue"
- "SignedByte"
- "SignedShort" (old "short integer")
- "SignedInteger" (old "integer")
- "SignedLong" (old "long integer")
- "UnsignedShort" (old "unsigned short integer")
- "UnsignedInteger" (old "unsigned integer")
- "UnsignedLong" ("unsigned long integer")
- "UnsignedByte" (old "byte")

(old type styles are also accepted)

fend [0, +]
byte swapping on/off (+fend: swap bytes, -fend: keep byte order (default))

fhbf [0, +]
HighByteFirst (default: machine byte order) overrides option fend

fras [1, D]
<DataRasterConfiguration> (default: 1)

fdvo [1, D]
<DataValueOffset> (default: 0)

ccd2saxs

Number of image blocks : 0

Arguments

ccd2saxs [options] <i/p file> <dummy> <first> <last> <inc> <o/p file> <o/p first> <i/p hm>

Options

ext [1, s]
extension of the edf output file (default: ".edf")

gas2saxs

Number of image blocks : 0

Arguments

gas2saxs [options] <i/p file> <dummy> <first> <last> <inc> <o/p file> <o/p first> <i/p hm>

Options

ext [1, s]
extension of the edf output file (default: ".edf")

gel2saxs

Number of image blocks : 0

Arguments

gel2saxs [options] <i/p file> <dummy> <o/p file>

saxs_add

Addition of two image sequences

Arguments

saxs_add [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

saxs_addcol

Average columns between col1 and col2 in image sequence 2 and add the results to each column of image sequence 1

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_addcol [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

Options

col1 [1, f]
first column

col2 [1, f]
last column

saxs_addrow

Average rows between row1 and row2 in image sequence 2 and add the result to each row of image sequence 1

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_addrow [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

Options

row1 [1, f]
first row

row2 [1, f]
last row

saxs_aff

Affine transformation of an image sequence

Tranformation matrix T

$$T = (A, B); A = \begin{pmatrix} T00 & T10 \\ T01 & T11 \end{pmatrix}; B = \begin{pmatrix} T20 \\ T21 \end{pmatrix};$$

$$W' = A * W + B$$

with coordinates W (input image), W' (output image)

see also: saxs_rot

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_aff [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2>
<t00> <t01> <t10> <t11> <t20> <t21>

Options

t00 [1, f]
transformation matrix element T00 (default: 1.0)

t10 [1, f]
transformation matrix element T10 (default: 0.0)

t20 [1, f]
transformation matrix element T20 (default: 0.0)

t01 [1, f]
transformation matrix element T01 (default: 1.0)

t11 [1, f]
transformation matrix element T11 (default: 0.0)

t21 [1, f]

transformation matrix element T21 (default: 0.0)

saxs_ascii

Transformation of an edf-file into ASCII format

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_ascii [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2>
<header> <swap> <head_1> <head_2>
Purpose: Transformation into ASCII format
Options: +-head +-swap +-hd1 +-hd2 -hdmk <marker> -ext <extension> +-noi +-waxs

Options

hist [1, +]
+-hist write/do not write history into header (default: +hist)

noi [1, +]
+-noi suppress/do not suppress image number extensions in output files (default: -noi)

waxs [1, +]
+-waxs calculate/do not calculate the exact scattering vector for large scattering angles. If the projection type is Saxs 2sin(Theta) is calculated from tan(2Theta) to calculated the scattering vector. If the projection type is Waxs this transformation is not done.
The option +waxs sets the reference system to -rsys saxs. (default: -waxs)

scf [1, f]
-scf <factor> : multiplication of coordinates with a scale factor, e.g. to tranform s in nm to q=2*pi*s in Angstrom (-scf "0.2*pi") (default: 1.0)

pref [1, s]
-pref 'prefix' : specify the prefix of the ouptput file

ext [1, s]
-ext 'extension' : extension of the output ascii file (default: ".txt")

head [0, +]
+-head : enables/disables the header printing

swap [0, +]
+-swap : enables/disables swap of rows and lines

hd1 [0, +]
+-hd1 : enables/disables the i_1 coordinate printing

hd2 [0, +]
+-hd2 : enables/disables the i_2 coordinate printing

abs [0, +]
+-abs : print absolut values of coordinates

spc [1, s]
+-spc : spacer between data columns, default: tabulator

saxs_ave

Averaging of two image sequences

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_ave [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

saxs_average

saxs_average [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <ofac> <ocon>

Calculation of the average and the variance of a series of n images. The number of images to add can be given with the option -add <n>. Without this option all input images are averaged. All data to be averaged will be copied into memory. This could fail for large series of images when not enough memory is available. In averaging mode "mean" the variances of the mean values are calculated and written into the error image (-omem -1). The option +var is set automatically and the variances of the calculated mean values are calculated by dividing the variances of each measurement by NM1 (default NM1=n-1, where n are the number of averaged pixels). NM1 can be set to n with the option -n-1 (default +n-1). Variance arrays of input images are not used.

The array sizes (Dim_1, Dim_2) of all input images must be identical. It is not possible to use reference systems and change reference system parameters, i.e. it is not possible to shift images or to use region of interests (ROI). To do this the input images must be tailored before the use of saxs_average, e.g. with saxs_mac.

Intensity averaging

In default mode the mean values and variances of all input images are calculated pixel by pixel. Dummy values are taken into account.

The averaging mode can be changed with the option -amod <mode>. The averages are done independently for each output pixel. The value of the output pixel is a function of the corresponding input pixels val[]:

-amod mean: calculate mean m and variance v of the mean: v=<(val[]-m)^2>/(n-1)

-amod median: calculate the median m and v as median((val[]-m)^2)/(n-1)

-amod quantil: calculate the pq-quantil m and v as pq-quantil((val[]-m)^2)/(n-1), the option pq must be used to change the quantil.

The default averaging mode is mean. The pseudo variances v for median and quantil are ad-hoc definitions. They can only be used as qualitative results.

Intensity filtering ("removal of cosmics")

In this mode the input pixel values are first classified and values outside the acceptance range are rejected. The remaining pixels are averaged in a second step.

The filter mode accepts the same functions as the averaging mode:

-fmod mean, -fmod median, -fmod quantil

The acceptance range in filter modes "mean" and "median" is

(2) [m-<sig->*sigma .. m+<sig+>*sigma]

The m-value is the average according to the averaging mode (-amod ...). The sigma value is the square-root of the variance (mean) or the corresponding pseudo-variance (median) per input pixel, i.e. without division by n-1.

The acceptance range can be adjusted with the options -sig- and -sig+. Their default values are 5.

The acceptance range in filter mode "quantil" is the interval between the lower and the upper quantil value, which can be set with the the options -loq <lower quantil> -upq <upper quantil>.

Options

-add [1, d]
-add <number of images to average> (default 0 to average all)

-amod
-amod <mean|median|quantil>: average function (default: mean)

-fmod
-fmod <mean|median|quantil>: filter function (default: none)

+/-n-1 [0, +]
+-n-1: calculate empiric variance (default +n-1)

-sig- [1, f]
-sig- <lowfac> : to exclude values smaller <lowfac>*sigma (default 5)

-sig+[1, f]
-sig+ <upfac>: to exclude values larger <upfac>*sigma (default 5)

-mins [1, f]
-mins <minimum sigma>: minimum sigma value (default 0.5)

-loq [1, f]
-loq <lower quantil> (default 0.1, filter mode quantil only)

-upq [1, f]
-upq <upq quantil> (default 0.9, filter mode quantil only)

-pq [1, f]
-pq <quantil p> : (default 0.5, averaging mode quantil only)

Examples:

Mean of all input images (no filtering):

- `saxs_average ka28_11_%%%%ccdraw,1,12 mean.edf`

Mean of all input images with median filtering:

- `saxs_average -fmod median ka28_11_%%%%ccdraw,1,12 mean1.edf`

Mean of all input images with 0.1-0.9 quantil filtering:

- `saxs_average -fmod quantil ka28_11_%%%%ccdraw,1,12 mean2.edf`

Median of all input images (no filtering):

- `saxs_average -amod median ka28_11_%%%%ccdraw,1,12 median.edf`

pq quantil of all input images (no filtering), a pq value of 0.5 (default) calculates the median:

- `saxs_average -amod quantil -pq 0.7 ka28_11_%%%%ccdraw,1,12 quantil.edf`

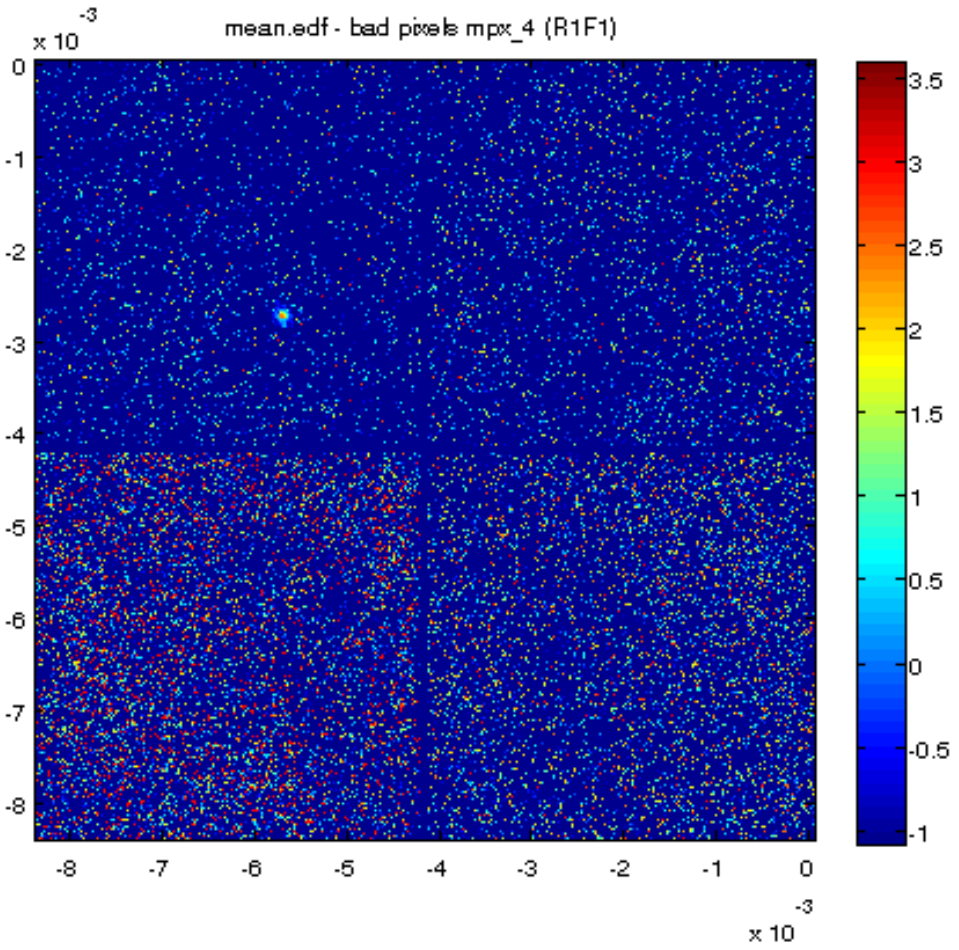

Fig. 3. Mean values of 12 images without filtering (mean.edf) and one hot spot.

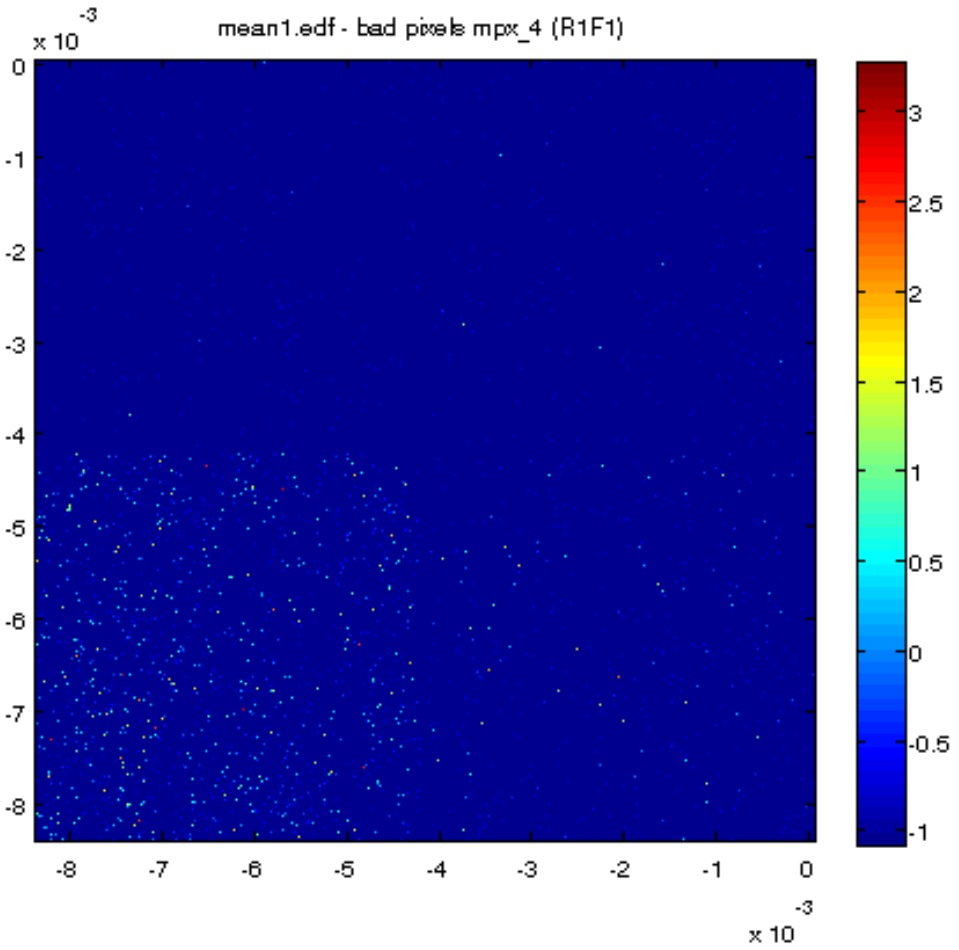

Fig. 4. Mean values of 12 images with median filtering (mean1.edf). To adjust the effect of filtering hot spots the option -sig+ <factor> can be used.

saxs_angle

Transformation of an image from cartesian coordinates to polar coordinates with pixel interpolation and azimuthal averaging.

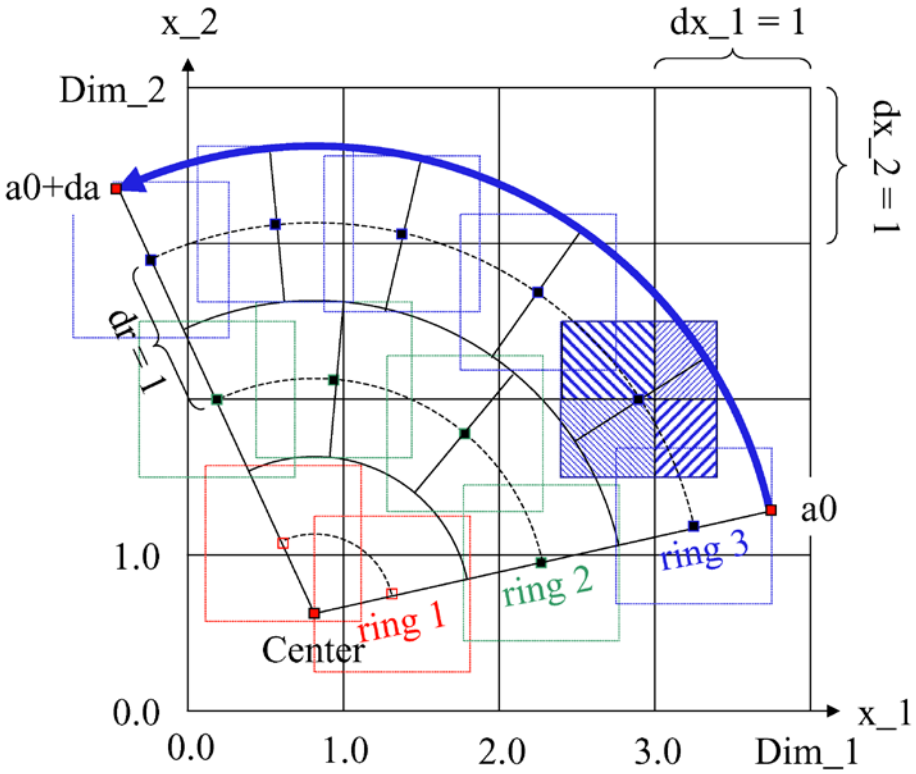

Fig. 5. Description of the azimuthal averaging procedure. The division of a section in several subsections is shown.

The integration is done on rings with thickness dr around the center. The default thickness is the minimum of the x_1 and x_2 pixel sizes in the chosen reference system. In the default reference system (Image) the default thickness is 1. In the following explanation it is assumed that the pixels are quadratic. The default values ensure that each pixel of the source image contributes to the result. Therefore, the radial default value (dr) should not be changed. The azimuthal default (da) can be changed.

Each ring is divided in subsections so that the azimuthal length of each subsection is less than the thickness dr. The integration uses the trapezoidal rule. Integration points are calculated for the center of each subsection by interpolation of the four neighbouring pixels. The weight of each pixel is proportional to the common cross section of a pixel with a squared pixel with size 1x1 around the interpolated point. (dashed pixel in Fig. 2).

x_1 of the output file corresponds to the radius

x_2 of the output file corresponds to the angle

The angle coordinate (x_2) uses always the real reference system (Offset, PixSiz). Center, WaveLength and SampleDistance are not used for coordinate 2 of the output image.

Version

4.10 2004-12-03

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_angle [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <odum>
<i1cen1> <i1cen2> <r0> <dr> <dim1> <a0> <da> <dim2>

Options

r0 [1, f]
-r0 <minimum radius>: default 0.0

dr [1, f]
-dr <radial interval>: default is a value that corresponds to 1 pixel

a0 [1, f]
-a0 <azimuthal start angle [radian]>: default 0.0_deg

da [1, f]
-da <azimuthal interval [radian]>: default 1_deg

s2 [0, +]
+-s2 This option is used to multiply the input data with the square of the radius (in rsys units) before transformation. This allows the direct calculation of the scattering power re^2*Q/V from the data by summation over i_1. (default: no multiplication)

The options asym and csym are used together with the option s2. They allow proper azimuthal averaging of intensities when the scattering pattern shows fiber symmetry and the fiber axis is lying in the detector plane. The angle of the fiber axis with the w_1 axis is defined by the option asym.
asym [1, f]
-asym <rotation of the symmetry axis relative to the w_1 axis [radian]>: default: 0.0_deg

csym [0, +]
+-csym switches on/off the multiplication of the input pattern with Pi/2*sin(Alpha-ASym), where Alpha is the angle of the data point from the w_1 axis and ASym the angle of the symmetry axis from the w_1 axis (default: no multiplication).

saxs_arc

Transformation of an image from cartesian coordinates to polar coordinates with pixel interpolation, including azimuthal averaging and radius/arc regrouping. This programs is a further development of the program saxs_angle. The usual options are defaults. With -i2nam <maskfile> the input image can be masked. The reference system for masking is always "region". In this way input image and mask can be differently binned, but the binning size must always be correctly defined.

Version

V4.30 2010-09-29

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

Usage: saxs_arc [options]

<i1nam> <onam> <i1fst> <i1lst> <i1inc> <odum>

<i1cen1> <i1cen2> <r0> <dim1> <a0> <da> <dim2>

For masking add the option "-i2nam <maskfile">

Options

arc [0, +]
+-arc switches from radius/angle to radius/arc regrouping (default -arc)

xy [0, +]
+-xy write xy-displacement files (default -xy)

xyna [1, s]
-xyna suffix of xy-displacement files (default -xyna file.edf)

r0 [1, f]
-r0 <minimum radius>: default 0.0

dr [1, f]
-dr <radial interval>: default is a value that corresponds to 1 pixel

-arc
a0 [1, f]
-a0 <azimuthal start angle [radian]>: default 0.0_deg

da [1, f]
-da <azimuthal interval [radian]>: default 1_deg

+arc
a0 [1, f]
-a0 <arc start [m radian]>: default 0.0_m_deg

da [1, f]
-da <arc interval [m radian]>: default (minimum of both pixel sizes)

s2 [0, +]
+-s2 This option is used to multiply the input data with the square of the radius (in rsys units) before transformation. This allows the direct calculation of the scattering power re^2*Q/V from the data by summation over i_1. (default: no multiplication)

The options asym and csym are used together with the option s2. They allow proper azimuthal averaging of intensities when the scattering pattern shows fiber symmetry and the fiber axis is lying in the detector plane. The angle of the fiber axis with the w_1 axis is defined by the option asym.
asym [1, f]
-asym <rotation of the symmetry axis relative to the w_1 axis [radian]>: default: 0.0_deg

csym [0, +]
+-csym switches on/off the multiplication of the input pattern with Pi/2*sin(Alpha-ASym), where Alpha is the angle of the data point from the w_1 axis and ASym the angle of the symmetry axis from the w_1 axis (default: no multiplication).

saxs_arc saxs_arc V4.30 2010-09-29, Peter Boesecke

saxs_bsl

Transformation of an edf image sequence into bsl format (daresbury otoko)

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_bsl [options]
<i1nam> <output header name> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2>

Options

hnam [1, s]
-hnam <output header name>: XNNMMM.EDF (default: automatically created from input filename)

bsl [0, +]
+bsl : write into a single bsl binary file (default: -bsl)

saxs_cave

Averaging of a point mirrored sequence with itself.

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_cave [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <center1> <center2>

Options

cen1 [1, f]
-cen1 <WCenter_1> : center 1 (input: user system coordinate, coordinate calculated according to first image)

cen2 [1, f]
-cen2 <WCenter_2> : center 2 (input: user system coordinate,coordinate calculated according to first image)

saxs_clip

Sets all values above maxclip to maxvalue and all values that are smaller than minclip to minvalue. Works like -min and -max on the read data (after transformation and binning), but the excluded pixels are set to minimum and maximum (not to dummy).

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_clip [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <micl> <macl>

Options

macl [1, f]
-macl : maximum

micl [1, f]
-micl : minimum

saxs_col

Projection of a vertical band in a sequence of images to a single column. The output column is swapped to a line. The swapped column is written to a line of the output image. Axis 1 of the output gets the axis type of input axis 2 of the first image of the sequence, axis 2 of the output image gets axis type numerator.

see also saxs_row

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_col [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <col1> <col2>

Version

V4.50 (2002-06-02)

Options

col1 [1, f]
first column of the vertical band (input: user system coordinate, coordinate calculated according to first image)

col2 [1, f]
last column of the vertical band (input: user system coordinate, coordinate calculated according to first image)

row1 [1, f]
first row of the input image (input: user system coordinate, coordinate calculated according to first image)

row2 [1, f]
last row of the input image (input: user system coordinate, coordinate calculated according to first image)

+/-vint [0, +]
multiply output with pixel size (+vint) (default: -vint)

Use the option -/+i1ave to select between averaging and summation of pixels. The option +/-vsum is not available any more.

saxs_curves

Write scattering curve to a text file.

Description

Writes scattering vector, intensity and sigma of a row-by-row saxs file. The projection type can either be Saxs or Waxs. Each line of the input image must correspond to an azimuthal average/projection, i.e. the scattering vectors of the n-th element in each line must be identical like for files created with saxs_angle, saxs_arc, saxs_row, saxs_col.

Arguments

saxs_curves [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <ofac> <ocon>

Options

-ext 'extension' : extension of the output file (default: ".txt")
-hdmk 'marker' : first character in a header line (default: "#")
-spc 'spacer' : spacer string between columns (default: "\t")
-hedl 'labels' : column labels, = is wildcard (default: s I stddev)

+-head : enables/disables the header printing (default: +head)
+-hd1 : enables/disables column labels (default: +hd1)
-scf <factor> : multiplication of scattering vector s (default: 1), -scf 2_pi converts s/nm to q/nm
+-hist : writes/do not write history into header (default:+hist)
+-waxs : calculate exact scattering vector length from saxs-coordinate (default: +waxs)

saxs_csub

Subtraction of a point mirrored sequence from itself.

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_csub [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <center1> <center2>

Options

cen1 [1, f]

-cen1 <WCenter_1> : center 1 (input: user system coordinate, coordinate calculated according to first image)

cen2 [1, f]
-cen2 <WCenter_2> : center 2 (input: user system coordinate, coordinate calculated according to first image))

saxs_div

Division of two image sequences. To calculate the inverse of each image in an image sequence with the name x.edf type: saxs_div -i1fac 0 -i1con 1 -i1nam x.edf -i2nam x.edf

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_div [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

saxs_divcol

Average columns between col1 and col2 in image sequence 2 and divide each column of image sequence 1 by the results

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_divcol [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

Options

col1 [1, f]
first column

col2 [1, f]
last column

saxs_divrow

Average rows between row1 and row2 in image sequence 2 and divide each row of image sequence 1 by the result

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_divrow [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

Options

row1 [1, f]
first row

row2 [1, f]
last row

saxs_filter

Remove spurious noise from detector data.

Description

Calculates a two dimensional gaussian with fwhm-width multiplied by the function (1.0-a*pow(2,(-16.0*(r^4/widtha^4)) which is zero in the center if a=1. The sum of the generated data points is normalized to 1. A should be positive. The resulting values for nearby pixels are written to a mask array. This array can be saved with the option -mnam <name>.

(1/mean) * exp(-r^2 * /(2*sigma^2)) * (1-a*2^(-16*r^4/widtha^4)

where 1/(2*sigma^2) = 4*log(2)/width^2

The value of each pixel in an image is compared with its neighbouring pixels weighted with the above equation. If its intensity differs more than max from the mean value and if its quadratic difference from the mean value is larger than chi2 the pixel is either set to dummy (mode 0) or replaced by the mean value (mode 1).

Arguments

saxs_filter [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <w> <wa> <a> <mrad> <chi2> <max>

Options

-mode <value> mode 0: set to dummy, 1: set to average
(default 0)

-chi2 <value> chi2 (default 5.0)
-max <value> maximum difference from mean value (default 0.0)
-w <value> gaussian width (default 4.0)
-wa <value> dip width (default 4.0)
-a <value> (default 1.0)

-rmsk <value> radius of mask (in pixel) (default 4)
-mnam <mask file name> mask file name (default "msk.edf")
+/-msk mask file flag (write/do not write a mask file)
(default -msk)

saxs_gauss

A pseudo-random number generator adds to each input pixel value I1 a random number ΔI that follows a gaussian distribution function with RMS distribution value σ:

$$(3) \quad G(\Delta I, \sigma) = \frac{1}{\sqrt{2\pi}\cdot\sigma} e^{-\frac{\Delta I^2}{2\sigma^2}}$$

The distribution value σ can be set with the option -sigm <sigm>. The seed of the pseudo-random number generator can be set with the option -seed <seed>. The seed is incremented and updated for each output image. The increment can be set with the option -sinc <increment>.

Calculation: I0 = GaussNoise(sigm)

Error propagation: V0 = V1+sigm*sigm

Number of image blocks : 2
Maximum number <n> of input blocks: 1

Arguments

saxs_gauss [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <ofac> <ocon> <sigm>
<seed> <sinc>

Options

sigm [1, F]
- sigma of the Gauss distribution.
- default: 0.0

seed [1, D]
- seed of the random number generator.
- default: 13131

sinc [1, D]
- increment of the random number seed for each new image
- seed' = (unsigned int) (seed + loopcnt * sinc)
  - loopcnt is internally increment for each image
- default: 2

saxs_gnaw

Sets pixels in an ellipse around a dummy also to dummy. Similar to the general option +ogna <gnc> <gnr>.

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_gnaw [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <gnc> <gnr>

Options

gnc [1, d]
horizontal radius

gnr [1, d]
vertical radius

saxs_log

Logarithm (base 10) of the image. To calculate the logarithm to another base multiply the output with -ofac "log(<base>)/log(10)"

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_log [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2>

saxs_mac

Mulitplication with a factor and addition of a constant to each image in a sequence. This routine performs a simple read and write of an image sequence. All general options are available and can be used to modify the values, e.g. to multiply with a factor and to add a constant: +ofac <fac> +ocon <con>. To convert an edf-file or bsl file to a standard edf-file with 512 byte block size and float binaries type saxs_mac old.edf new.edf or saxs_mac A01000.BSL new.edf.

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_mac [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <ofac> <ocon> <shft1> <shft2>

saxs_mul

Multiplication of two image sequences.

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_mul [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

saxs_mulcol

Average columns between col1 and col2 in image sequence 2 and multiply each column of image sequence 1 with the results

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_mulcol [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

Options

col1 [1, f]
first column

col2 [1, f]
last column

saxs_mulrow

Average rows between row1 and row2 in image sequence 2 and multiply each row of image sequence 1 with the result

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_mulrow [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>

<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

Options

row1 [1, f]
first row

row2 [1, f]
last row

saxs_new

Create a saxs data file using arithmetical expressions. The options are -val <expression> and -err <expression>. The expressions after -val and -err accept as variables _x1 for coordinate 1 and _x2 for coordinate 2 in the chosen reference system. The expression after -err excepts also the variable _I for intensity of the val array.

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Arguments

saxs_new [options]
<onam> <odum> <odim1> <odim2> <val>
Generates an image from an expression(_x1,_x2)

saxs_mul

Multiplication of two image sequences.

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_mul [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

saxs_normn

Normalization of an image sequence to absolute intensities (+ccd for ccd, -ccd for delay line detector, default: -ccd)

Number of image blocks : 4

Maximum number <n> of input blocks: 3

Version

2004-10-31

Arguments

saxs_normn [options]
<i/p file> <scaler file> [<flatfield file>] <o/p file>
<first image> <last image> <inc>
<first scaler header> <inc scaler header>
[<flatfield image>]
<o/p dummy> <o/p dim1> <o/p dim2>
<transmission>
<scaler I0 mon.> <scaler exposure time> <scaler anode counts>
<2nd scaler I0 mon.> <2nd scaler exposure time> <2nd scaler anode counts>

General Description

The program saxs_normn normalizes a two dimensional scattering pattern measured with a plane two-dimensional detector to absolute intensities in units of scattered photons per steradian and per incident photon. In other words, it is the scattering cross section $d\sigma/d\Omega$ per sample cross section A. To correct for absorption the normalized pattern is divided by the sample transmission T. The last step is optional. The transmission corrected scattering cross section per sample cross section can be calculated in the following way:

$$(4) \quad \frac{1}{T}\cdot\left(\frac{1}{A}\cdot\frac{\partial\sigma}{\partial\Omega}\right)=\frac{1}{T}\cdot\left(\frac{i_s}{I_0}\cdot\frac{r_d^{\,2}}{p_1\cdot p_2}\cdot\frac{r_d}{L}\right)$$

I_0 is the number of incident photons during the exposure, i_s is the number of detected photons in a single pixel with the area $p_1 \bullet p_2$, r_d is the distance between sample and pixel and L the length of the detector normal to the sample (keyword SampleDistance). The division by the spherial angle $d\Omega$ is done by the last two factors in formula (1). The factor r_d/L corrects the projection of the pixel area to the scattering sphere under large angles. To calculate the scattering cross section per scattering volume $d\Sigma/d\Omega$ of a non-absorbing sample the result must be divided by the effective sample thickness d_{eff}.

$$(5) \quad \frac{\partial\Sigma}{\partial\Omega}=\frac{1}{T}\cdot\left(\frac{1}{A}\cdot\frac{\partial\sigma}{\partial\Omega}\right)\cdot\frac{1}{d_{eff}}$$

The division by the transmission T is effectively an absorption correction for a flat sample with thickness d_{eff}. In small angle X-ray scattering the transmission and the effective sample thickness are practically independent of the scattering angle. This is usually not the case in wide angle X-ray scattering. In this case the X-ray attenuation length and the sample shape influence the scattering pattern and formula (2) is no longer valid. In this case saxs_normn can still be used to calculate the number of scattered photons per steradian per incident photon, a property that can be measured without any knowledge about the scattering sample.

To take into account a non-homogeneous response over the detector area input pattern can be divided by a flat-field pattern F. A flat-field pattern is the detector response to a homogeneous illumination.
If extinction is small the transmission T can be approximated by I1/I0, where I0 is the number of photons in the incident primary beam during the exposure and I1 is number of photons in the transmitted primary beam.

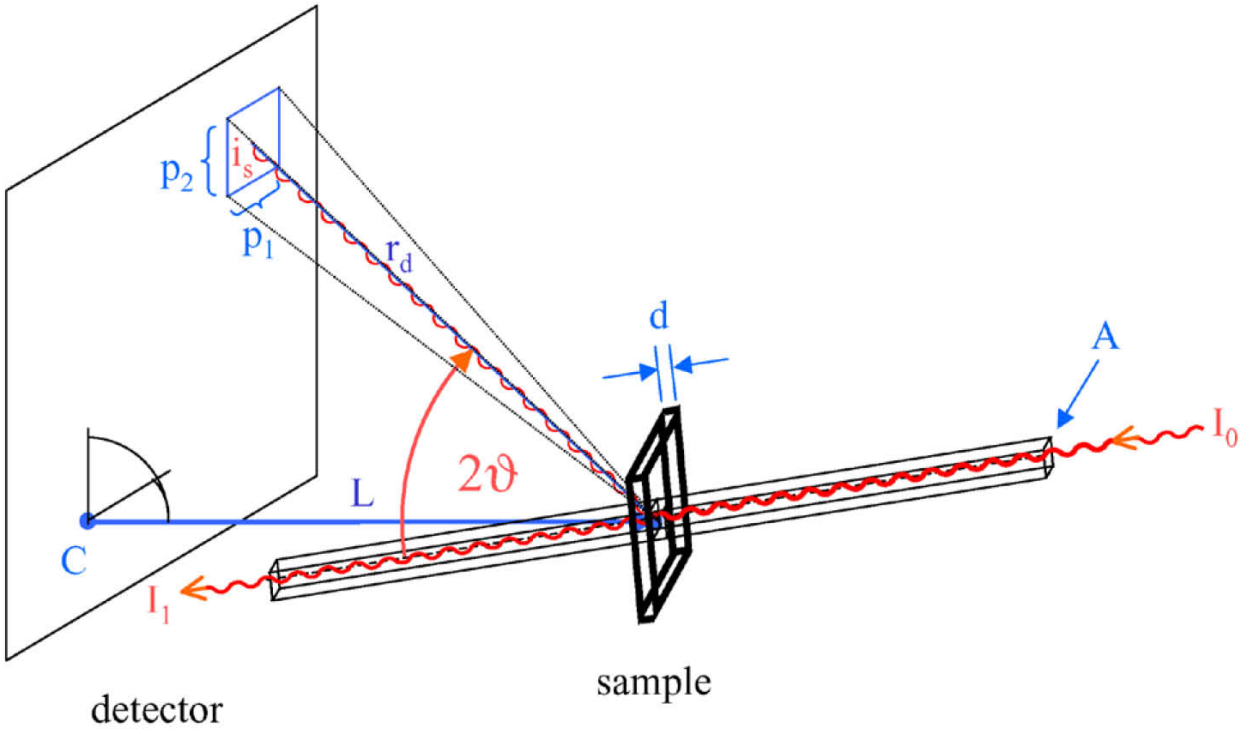

Fig. 6. Scattering geometry used in saxs_normn. I_0 (Intensity0) is the number of photons in the incident beam with cross section A, I_1 (Intensity1) is the number of photons in the transmitted beam. The number of scattered photons is i_s. L (SampleDistance) is the smallest distance between sample and detector. C (Center_1, Center_2) is the corresponding point on the detector. The pixel size is p_1*p_2 (PSize_1, PSize_2). The distance between sample and detecting pixel is r_d. A*d is the scattering volume, where d is the thickness of the sample. The incident beam does not need to be perpendicular to the detector surface.

Method

The program saxs_normn divides each pixel by the intensity I_0 of the incident beam (option -monv <Intensity0>) and by the sample transmission. As default the transmission is equal to Intensity1/Intensity0, where Intensity1 is the number of transmitted photons (option -trmv <Intensity1>). Intensity0 and Intensity1 are read from the file header. The default corresponds effectively to a normalizion to Intensity1 because the Intensity0 cancels out. Alternatively, a fixed transmission value can be entered (option -trm <tranmission>). In this case, I_1 is equal to T*I_0. In this case, the header value Intensity1 is ignored. The pixel intensity i_s is divided by the spherical angle $\Delta\Omega$ under which it observes the scattering.
To take into account the different sensitivities of the pixels the scattering pattern can be divided by a flat-field image F. In this case the option +flat must be set (default -flat) and the name of the flat-field image must be given after the option -i3nam <flat-field>.
The calculation of the transmission corrected scattering cross section per sample cross section is done in the following way:

$$(6) \quad \frac{1}{T}\cdot\left(\frac{1}{A}\cdot\frac{\partial\sigma}{\partial\Omega}\right)=\frac{1}{T}\cdot\left(\frac{i_s}{Intensity0}\cdot\frac{1}{\Delta\Omega}\right)$$

$$(7) \quad i_s=\frac{I(i\_1,i\_2)}{F(i\_1,i\_2)}\cdot f$$

i_s is the number of scattered photons at pixel coordinate (i_1, i_2), I(i_1, i_2) is the detector pixel intensity, F(i_1, i_2) is the flat-field value (+flat -i3nam <flat-field file> and f a normalization factor, that can be entered by the option -i1fac <f>)

$$(8) \quad T=\frac{Intensity1}{Intensity0}$$

T is the sample transmission, it can be modified with the option -trm <T>, Intensity0 can be modified with the option -monv <Intensity0> and Intensity1 can be modified with the option -trmv <Intensity1>

$$(9) \quad \Delta\Omega=\frac{PSize\_1\cdot PSize\_2}{r_d{}^2}\cdot\frac{SampleDis\tan ce}{r_d}$$

$\Delta\Omega$ is the spherical angle under which the detector pixel sees the scattering volume, PSize_1 and PSize_2 are the pixel sizes (option -i1pix <PSize_1> < PSize_2>), SampleDistance is the distance between detector and sample (option -i1dis <SampleDistance>). r_d is the distance between the detector pixel and the scattering volume.

$$(10) \quad r_d=\sqrt{SampleDis\tan ce^2+\left(i\_1+Offset\_1-Center\_1\right)^2+\left(i\_2+Offset\_2-Center\_2\right)^2}$$

(i_1,i_2) is the pixel coordinate, Offset_1 and Offset_2 are the coordinate offsets and (Center_1, Center_2) is the pixel coordinate of the point of normal incidence on the detector.

Call

The basic call to normalize a 2d CCD scattering pattern is:

a) without flat field

- `saxs_normn -p +ccd -i1nam <in> -onam <out>`

b) with flat field

- `saxs_normn -p +ccd +flat -i3nam <flat-field> -i1nam <in> -onam <out>`

The input file header must contain the following keywords. It they are missing the values must be given with the option shown in the second column:

Intensity0	-monv <Intensity0>
Intensity1	-trmv <Intensity1>
ExposureTime	-timv <ExposureTime>
Center_1, Center_2	-i1cen <Center_1> <Center_2>
PSize_1, PSize_2	-i1pix <PSize_1> <PSize_2>
SampleDistance	-i1dis <SampleDistance>

The parameter ExposureTime value is only required to display the detector summary. It is not used to calculate the output pattern. If not known, it can be set to 1. The displayed detector summary may be wrong.
If the input file does not contain the values Intensity0, Intensity1 and ExposureTime, they can be read from a second input file that contains the header values, e.g. the raw data file.

- `saxs_normn -p +ccd -i1nam <in> -i2nam <header-file>-onam <out>`

Data from a delay line detector

If the normalization is applied to data from a delay line detector the intensities can also be corrected for the dead time of the time to digital converter (TDC). In this case, the sum of the anode counts must be measured and given in the header after the keyword Anode (-anov <Anode>). The pixel intensities I(i_1,i_2) of the detector are scaled by the factor Anode/Sum(I(i_1,i_2)). In case of no losses the ratio should be 1. This correction does only work when the whole detector pattern is used, when the rejection is independent of the position on the detector and as long as the anode count rate is proportional to the global scattering intensity. Only global count rate losses are corrected.

Anode	-anov <Anode>

The basic program call is:

- `saxs_normn -p -ccd -i1nam <in> -onam <out>`

Multichannel scaler

It is possible to read the intensity information (Intensity0, Intensity1, Anode, ExposureTime) from a multi channel scaler (MCS). Normally, the program saxs_normn tries to find the multi channel scaler keywords HSI0, HSI1 and HSTime (and HSAnode in case of a delay line detector). If they are not found saxs_normn looks automatically for the keywords Intensity0, Intensity1, ExposureTime and Anode. The MCS counts of scaler nn is written after the keyword HS32Cnn, the zero value after the keyword HS32Znn, the calibration factor after the keyword HS32Fnn and its name after the keyword HS32Nnn. The scaler values are calculated in the following way:

(11) value(nn,time) = (HS32Cnn-HS32Znn*time)*HS32Fnn

and specifically:

(12) ExposureTime = value(HSTime,0)

(13) Intensity0 = value(HSI0, ExposureTime)

(14) Intensity1 = value(HSI1, ExposureTime)

(15) Anode = value(HSAnode,ExposureTime)

To force saxs_normn not to read from scalers, the option -scal must be given.

Options

trm [1, f]
transmission (0..1)

dpth [1, f]
eff. scaler depth in bits,
e.g. scaler depth=24 and 4 accumulations
=> -dpth "24+log(4)/log(2)" = 26

monc [1, f]
absolute counts of I0 scaler

monz [1, f]
zero value of I0 scaler

monf [1, f]
factor of I0 scaler (conversion of counts to photons)

monv [1, f]
number of incident photons

timc [1, f]
absolute counts of time scaler

timf [1, f]
factor of time scaler (for conversion of counts to seconds)

timv [1, f]
exposure time in seconds

anoc [1, f]
absolute counts of anode scaler

anof [1, f]
factor of anode counter (for conversion of counts to anode counts, usually 1)

anoz [1, f]
zero value of anode scaler

anov [1, f]
anode value in counts

trmc [1, f]
absolute counts of I1 scaler

trmz [1, f]
zero value of I1 scaler

trmf [1, f]
factor of I1 scaler (conversion of counts to photons)

trmv [1, f]
number of transmitted photons

mo2c [1, f]
absolute counts of second I0 scaler

mo2f [1, f]
factor of second I0 scaler (conversion of counts to photons)

ti2c [1, f]
absolute counts of second time scaler

ti2f [1, f]
factor of second time scaler (for conversion of counts to seconds)

an2c [1, f]
absolute counts of second anode scaler

an2f [1, f]

factor of second anode scaler (for conversion of counts to anode counts)

tr2c [1, f]
absolute counts of second I1 scaler

tr2f [1, f]
factor of second I1 scaler

sum [1, f]
sum of input image 1. This option can be used to set the sum of the input image externally.

bcf [1, f]
The option -i1bin a b averages over a*b pixels. To calculate afterwards the correct rejection the binning correction factor must be set to -bcf "a*b". The binning correction factor cancels out and has no effect on the output image data, but only on the displayed acceptance and rejection values.

rdis [1, f]
relative distance of sample from the home position of the detector displacement. If RelDis is set, the value of SampleDistance is ignored and replaced by the sum of DetectorPosition plus RelDis in input sequence 1 and 2 (i1 and i2) and in the output sequence (o). The value of DetectorPosition is read from input sequence 2.

mons [1, d]
number of I0 monitor scaler

tims [1, d]
number of exposure time scaler

anos [1, d]
number of anode counts scaler

trms [1, d]
number of I1 monitor scaler

mo2s [1, d]
number of second I0 monitor scaler

ti2s [1, d]
number of second exposure time scaler

an2s [1, d]
number of second anode count scaler

tr2s [1, d]
number of second I1 monitor scaler

slen [1, d]
scaler length total number of available scalers (1..slen)

flat [0, +]
flatfield correction/no floatfield correction

tron [0, +]
- +tron: use user supplied tranmission value (option: -trm <transmission>)
- -tron: calculate transmission with scaler values (I1/I0)
- default: -tron

scal [0, +]
- +scal: use scaler values to calculate I0, I1, exposure time and anode counts.

- -scal: read I0, I1, exposure time and anode counts from keywords in the file
- default +scal, automatically set to -scal if scaler information not found

`ccd` [0, +]
- +ccd : no rejection correction
- -ccd : rejection correction (default)

`mon` [0, +]
- +mon : normalization to monitor intensities (default)
- -mon : no normalization to monitor intensities

`dis` [0, +]
- +dis : distance normalization (normalization to spherical angles) (default)
- -dis : no distance normalization

saxs_patch

Replacement of the first image by the second image (where it is defined). The second image is patched into the first image.

Number of image blocks : 3
Maximum number <n> of input blocks: 2

Arguments

saxs_patch [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2>

Options

ex [0, +]
- +ex: do not copy dummy values from image 2 to image 1
- -ex: copy all pixels from image 2 to image 1
- default: -ex

dumo [0, +]
- +dumo: only the dummy values of image 1 are patched with the values of image 2
- -dumo: replace all values of image 1
- default: -dumo

saxs_poisson

A pseudo-random number generator replaces each input pixel value I1 with a random number k that follows Poisson statistics by using I1 as the expectation value ν:

$$(16) \qquad P(k,\nu)=\frac{\nu^k}{k!}e^{-\nu}$$

P(k,ν) is the probability to find during any measurement k events, where ν is the expectation value of events for an infinite number of measurements.

For faster calculation a gaussian distribution is used for I1 values above 100 by using sqrt(ν) for σ.

(17) $$P(k,v) \approx G(k,v) = \frac{1}{\sqrt{2\pi \cdot v}} e^{-\frac{k^2}{2v}}$$

The seed of the pseudo-random number generator can be set with the option -seed <seed>. The seed is incremented and updated for each output image. The increment can be set with the option -sinc <increment>.

Calculation: I0 = PoissonNoise(I1)

Error propagation: V0 = V1+I1

Number of image blocks : 2
Maximum number <n> of input blocks: 1

Arguments

saxs_poisson [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <ofac> <ocon> <seed> <sinc>

Options

seed [1, D]
- seed of the random number generator.
- default: 31313

sinc [1, D]
- increment of the random number seed for each new image
- seed' = (unsigned int) (seed + loopcnt * sinc)
  - loopcnt is internally increment for each image
- default: 2

saxs_pol

Division/multiplication by the polarization factor. All angles are in radian or must be followed by a unit, e.g 30_deg, 10_mrad.

The polarization factor P is splitted into an unpolarized P(1) and a polarized P(2) part:

(18) P = P(1) + P(2)

(19) P(1) = (1-Pol) * (1 + kout3*kout3)/2

(20) P(2) = Pol * (
(1-kout1*kout1) * 0.5 * (1+cos(2*PChi)*cos(2*PPsi))
+ (1-kout2*kout2) * 0.5 * (1-cos(2*PChi)*cos(2*PPsi))
- kout1*kout2 * cos(2*PChi)*sin(2*PPsi))

The scattered beam kout is either calculated from an Ewald-sphere projection (ProjectionType = Waxs) or from a detector image (ProjectionType = Saxs). The calculation of kout is done with the same routines that are used in saxs_waxs. The rotation parameters etc are identical. Rotations are only used for detector images (ProjectionType = Saxs). The options -i1pro Saxs or -i1pro Waxs set the projection type manually.

The polarization state is described by the degree of polarization Pol and the Poincaré sphere: PChi (ellipticity) and PPsi (polarization direction).

Number of image blocks : 2
Maximum number <n> of input blocks: 1

Version

V4.52 2004-10-14

Arguments

Usage: saxs_pol [-mul]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <odum> <odim1> <odim2>
<pol[1.0]> <pchi[0.0]> <ppsi[0.0]>

Options

pol [1, F] <Pol>
degree of polarization (0<=Pol<=1)
pchi [1, F] <PChi>
ellipticity (radian, -pi/4<=PChi<=+pi/4)

- PChi=-pi/4 left hand (cw) circular polarization
- PChi<0 left hand polarization
- PChi==0 linear polarization
- PChi>0 right hand polarization
- PChi=pi/4 right hand (ccw) circular polarization

ppsi [1, F] <PPsi>
rotation of the polarization ellipse/axis (radian, 0<=PPsi<pi).
The angle PPsi is measured ccw around axis x_3 starting at axis x_1
i1pro [1, P] <pro> (general option, automatically set by image header)
Apply the calculation to a Saxs image (i1pro Saxs) or to a Waxs projection (i1pro Waxs).
i1rot [1, F] <i1rot1> <i1rot2> <i1rot3> (general option, define detector rotation angles)
<i1rot1>, <i1rot2>, <i1rot3>: detector rotation angles (radian).
mul [1, +]
Multiply input image with polarization factor instead of dividing it

saxs_power

Calculate the n-th power of each image value

Number of image blocks : 2
Maximum number <n> of input blocks: 1

Arguments

saxs_power [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <power>

Options

pow [1, f]

power exponent

saxs_refract

Refraction correction of 2d-sub-surface scattering patterns. The calculation is valid for grazing incidence small angle scattering (GISAXS) and higher angle scattering. The incoming beam is refracted at the surface, scattered below the surface inside the sample and again refracted at the surface before it hits the detector. This routine transforms the observed (external) scattering pattern into the internal scattering pattern that would be observed inside the sample below the surface. The calculation is done using Snell's law. The surface normal N^ is defined by the angles chi and eta in the following way:

(21) N^ = (cos(chi),
cos(eta)*sin(chi),
sin(eta)*sin(chi))

The primary beam is pointing against axis3 = axis1 X axis2. The surface normal N^ is initially pointing along axis1. It is first rotated ccw around axis3 (with chi). Then N^ is rotated around axis1 (with eta).

Version

V4.05 2004-10-14

Arguments

saxs_refract [options] <i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <delta[0.0_deg]> <the[0.0_deg]> <chi[90.0_deg]>

Options

del [1, F]
1-delta : real part of the refractive index (0<=delta<1, default 0.0)

chi [1, F]
rotation of the sample around the primary beam in radian (corresponds to angle chi on id01) (radian or '_' and unit, 0_deg<=chi<=90_deg, default 90_deg). Chi is pi/2 (90_deg) when the sample surface is horizontal.

eta [1, F]
inclination of the chi-circle around axis 1 in radian (corresponds to angle eta on id01) (0_deg<=eta<90_deg, default 0.0_deg)

saxs_rot

Rotation of an image sequence by alfa in radian

$$T = (A, B); A = \begin{pmatrix} \cos(alfa) & -\sin(alfa) \\ \sin(alfa) & \cos(alfa) \end{pmatrix}; B = \begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix};$$

$$W' = A * W + B$$

with coordinates W (input image), W' (output image)

see also: saxs_aff

Number of image blocks : 2

Maximum number <n> of input blocks: 1

Version

V4.04 2004-10-14

Arguments

saxs_rot [options] <i1nam> <onam> <i1fst> <i1lst> <i1inc> <odum> <odim1> <odim2> <alfa>

Options

-alfa [1, f]

ccw-rotation of the image-in radian. The rotation center is the origin of the chosen reference system.

saxs_row

Projection of a horizontal band in a sequence of images to a single row. The row is written to a line of the output image. Axis 1 of the output gets the axis type of input axis 1 of the first image of the sequence, axis 2 of the output image gets axis type numerator.

see also: saxs_col

Number of image blocks : 2
Maximum number <n> of input blocks: 1

Version

V4.50 (2002-06-02)

Arguments

saxs_row [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <row1> <row2>

Options

col1 [1, f]

first column of the input image (input: user system coordinate, coordinate calculated according to first image)

col2 [1, f]

last column of the input image (input: user system coordinate, coordinate calculated according to first image)

row1 [1, f]

first row of the horizontal band (input: user system coordinate, coordinate calculated according to first image)

row2 [1, f]

last row of the horizontal band (input: user system coordinate, coordinate calculated according to first image)

+/-vint [0, +]
multiply output with pixel size (+vint) (default: -vint)

Use the option -/+i1ave to select between averaging and summation of pixels. The option +/-vsum is not available any more.

saxs_scal

Read the value of a keyword from the headers of each picture and write it to a text file.

Number of image blocks : 2
Maximum number <n> of input blocks: 1

Arguments

saxs_scal [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <keyword>

Options

key [1, s]
Keyword

pref [1, s]
Prefix of the output file

ext [1, s]
Extension of the output file

fnam [1, s]
Name of the output file

saxs_stat

Calculates statistics of the input image and write the result into a text file.

Number of image blocks : 2
Maximum number <n> of input blocks: 1

Arguments

saxs_stat [options]
<i1nam> <i2nam> <o/p prefix> <i1fst> <i1lst> <i1inc>
<selection>

Options

sel [1, s]
selection

- inum Image number

- dim1 Dimension 1
- dim2 inferior limit 1
- sup1 superior limit 1
- inf2 inferior limit 2
- sup2 superior limit 2
- ntot Number of data points
- nval Number of valid data points
- ndum Number of dummies
- max Maximum Value
- min Minimum Value
- amin Minimum absolute value
- dum Dummy value
- sum Sum of data values
- ssum Sum of squared data values
- mean Mean value = sum / nval
- smean Mean squared value = ssum / nval
- rms Root mean square value = sqrt(smean)
- devi Standard deviation value = smean - mean*mean
- gc1 Center of gravity 1

pref [1, s]
Prefix of the output file

ext [1, s]
Extension of the output file

fnam [1, s]
Name of the output file

show [0, +]
Print results to the standard output

saxs_sub

Subtraction of two image sequences

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_sub [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

saxs_subcol

Average columns between col1 and col2 in image sequence 2 and subtract the results from each column of image sequence 1

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_subcol [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

Options

col1 [1, f]
first column

col2 [1, f]
last column

saxs_subrow

Average rows between row1 and row2 in image sequence 2 and subtract the result from each row of image sequence 1

Number of image blocks : 3

Maximum number <n> of input blocks: 2

Arguments

saxs_subrow [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc> <i2fst>
<odum> <odim1> <odim2> <i1con> <i1fac> <i2con> <i2fac>
<ocon> <ofac>

Options

row1 [1, f]
first row

row2 [1, f]
last row

saxs_tiff

Transformation of an edf image sequence to 1byte/2byte/4byte tiff files. If more than 1 image is read from the input file the output files are numbered. The input file is first converted to an intermediate output file and then converted to a tiff file. The output dummy is never scaled. The title is written as comment into the tiff file. The outptu byte order is given by the machine byte order.

Hint: To convert a 16 bit unsigned integer edf-raw data file to a tiff files with 16 bps the options "-auto -bps 16" should be set. If the file contains dummy values the output dummy value should be set to a positive value inside the mapping range. If it lies outside the output range it is forced to the closest number inside the output range.

Number of image blocks : 2
Maximum number <n> of input blocks: 1

Arguments

saxs_tiff [options]

<i1nam> <tiff file name -tnam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <8|16|32-bps> <1-8-rori>
<auto> [<min> <max>]

Options

bps [1, d]
bits per sample in output tiff image
<8|16|32> : 8, 16 and 32 bits in tiff image (default 16)

rori [1, d]
orientation of the output tiff image
1: (x,y)
2: (-x,y) x-axis inverted
3: (x,-y) y-axis inverted (default)
4: (-x,-y) both axes inverted
5: (y,x) axes swapped
6: (y,-x) axes swapped and x-axis inverted
7: (-y,x) axes swapped and y-axis inverted
8: (-y,-x) axes swapped and both axes inverted

auto [1, +]
-auto: As usual, the output dummy is set to -odum or its default. If not explicitely set by -min or -max, minimum and maximum of the input image are set 0 and pow(2,bps)-1, all values in the range [minimum, maximum] are mapped to [0,(pow(2,bps)-1)]
+auto: The output dummy is set to upper range value (pow(2,bps)-1). If not explicitely set by -min or -max, minimum and maximum of the input image are calculated, all values in the range [minimum, maximum] are mapped to [0,(pow(2,bps)-2)] (default +auto)

-auto should be used for format transformations, +auto for image display, e.g. with xv.

min [1, f]
minimum of mapping range, values smaller or equal to min will be set to 0, all values in the range [minimum, maximum] are mapped to [0,(pow(2,bps)-1)]

max [1, f]
maximum of mapping range, value larger or equal to max will be set to pow(2,bps), all values in the range [minimum, maximum] are mapped to [0,(pow(2,bps)-1)]

Example

saxs_tiff saxs_tiff 3.0 1999-11-12, Peter Boesecke

Input sequence 1 [input.edf] : flat.new230696

Output tiff file [flat.tif] :

First image of input sequence 1 [1] :
Last image of input sequence 1 [1] :
Increment of input sequence 1 [1] :
Output dummy [-1.000000e+00] :
Output dimension 1 [512] :
Output dimension 2 [512] :
Bits per sample (8, 16, 32) [16] :
Orientation of output image (1-8) [3] :
Range output image with minimum and maximum? [FALSE] :
Minimum value [0.000000e+00] :
Maximum value [6.553500e+04] :

input file : flat.new230696
output prefix : flat.tif
first image : 1
last image : 1
increment : 1
output dummy : -1.000000
output dimension 1 : 512
output dimension 2 : 512
bits per sample : 16
autorange : 0
minimum value : 0.000000
maximum value : 65535.000000

Reading image: flat.new230696, Image : 1 done

Calculating ihb[0, 1] = Function(ihb[1, 1])

Input file : flat.new230696
Input image number : 1
Output tiff file : flat.tif

End of /home/boesecke/programs/saxs/bin/saxs_tiff

saxs_waxs

Projection of the Ewald sphere measured with a flat area detector (projection type Saxs) to a plane perpendicular to the incident beam (projection type Waxs) and vice versa. The detector can be inclined. Additionally, xy-displacement files can be written that can be used by the spatial distortion program spd. The projection will be called WAXS-transformation (see Fig. 4).

The length of the radial coordinate in the projected pattern corresponds to the length of the scattering vector s. The azimuthal angle corresponds to the azimuth of s. The projection is strictly valid for isotropic scattering patterns. In all other cases it is only a useful projection.

The scattering vectors are calculated using saxs-coordinates (default: -rsys saxs). The keywords PSize_N, Center_N, SampleDistance and WaveLength are required. The detector rotation can be defined with the option -i1rot <i1rot1> <i1rot2> <i1rot3>.

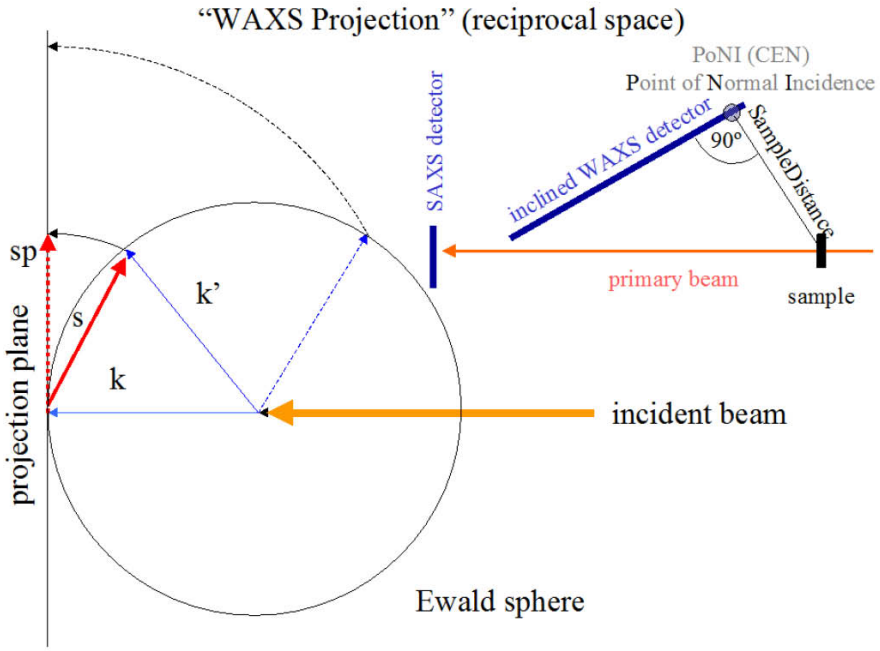

Fig. 7. Illustration of the WAXS projection. Pixel distances di on a Saxs projection are proportional to distances dx on the detector: dx = di * PSize. Pixel distances di on a Waxs projection are proportional to distances ds in reciprocal space (WaveLength0=1e-9 for convenience, sp=2*sin(Theta)*(WaveLength0/WaveLength): dsp = di * (PSize/SampleDistance) * (WaveLength0/WaveLength)

Version

V4.12 2004-10-14

Arguments

saxs_waxs [options] <i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <i1rot1> <i1rot2> <i1rot3>

Options

i1rot [1, F] <i1rot1> <i1rot2> <i1rot3> (general option, define detector rotation angles)
<i1rot1>, <i1rot2>, <i1rot3>: detector rotation angles (radian).
i1pro [1, P] (general option, automatically read from header)
projection type of the input image (default: saxs)
The input projection type is read from the header of the input image. Its default is saxs. This option allows to change the input value.
opro [1, P] (general option, preset to waxs)
required projection type of the output image (default: waxs)
The output projection type is set to waxs if it is not changed by the option opro. An input image that is already in the projection type waxs will not be transformed again. To transform this image back to a detector pattern the projection type of the output image must be set to saxs: -opro saxs.
xy [1, +]
write/do not write xy-displacement files (default: -xy)
Xy-displacement files can be used as input for the spatial distortion correction program spd, e.g. spd xfile=xfile.edf yfile=yfile.edf.

An xy-displacement image contains for each input pixel the difference between the pixel position in the output array and the pixel position in the input array. The x-displacement file contains the displacements in x_1 direction, the y-displacement file contains the displacement in the x_2 direction. The xy-displacement images are defined for the input image. The displacement images have the same dimensions as the input image and they are described by the same geometrical header parameters.

The geometrical header parameters of the output image are written to the following keywords:

DisplacedBsize_1, DisplacedBsize_2
DisplacedCenter_1, DisplacedCenter_2,
DisplacedOffset_1, DisplacedOffset_2,
DisplacedProjectionType,
DisplacedPsize_1, DisplacedPsize_2
DisplacedSampleDistance, DisplacedWaveLength

They can be used by the spatial distortion program to write the header of the output image.

xyna [1, s]
base name of xy-displacement files (default: file). The xy-displacement files are called x<xyna>.edf and y<xyna>.edf, e.g. xfile.edf and yfile.edf.

Example

saxs_waxs [options] <i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <i1rot1> <i1rot2> <i1rot3>

Projection of a SAXS pattern to the Ewald-sphere

rapid2saxs

Conversion of rapid bsl files to edf files (frames, scalers, timing). The program uses the standard edfio routines to read rapid files.

It copies rapid data to edf data files including scaler data:

The general option -add <number of frames> allows summation over multiple frames. The scaler data are also summed. The timing data is also modified, the deltatime of the first added frame is used, the new exposure time is the sum of the exposure times of all added frames. rapid2saxs can also be used to average a sequence of edf-detector file and its header data (HS32Cnn, Intensity0 and Intensity1), e.g. a ccd file[+].

Example:

BSL data consist of a header file and a number of memory files. Each memory file contains three-dimensional binary data (rows, colums, frames).

The file name definition is very strict. Header and memory files must comply with the following naming convention (C: ASCII character, M, N: number between 0 and 9).
Header file: CNN000.CCC
Memory file: CNNMMM.CCC

A rapid data set can consist of up to 4 files:
A01000.323 (header file, readable ASCII data)
A01001.323 (memory 1: binary image data, contains all frames in a single file)
A01002.323 (memory 2: binary scaler data, optional)
A01004.323 (memory 3: binary timing data, optional)

The header file defines the link between a file and a memory number. In the above example the header file looks like this:

```
SRS Data recorded on Sun Mar 23 15:43:27 2003
256 frames moving x-rays
     512     512     256       0       0       0       0       0       0       1
A01001.323
     256       1       1       0       0       0       0       0       0       1
A01002.323
     256       4       1       0       0       0       0       0       0       0
A01004.323
```

The two first lines (maximum 80 characters) contain a description (free format).

The third and forth lines contain the format (rows, columns, frames) and the filename of memory 1, here A01001.323. The file contains 256 frames of 512 x 512 pixels. Each number consists of exactly 8 characters.

Each following successive line pair defines in the same way an additional memory. The last number is 1 except for the last memory definition.

The scaler memory (memory 2) contains 1 frame, 256 rows, each containing a single scalar value.

The timing memory (memory 3) contains 1 frame, 256 rows, each containing 4 scalars (dead time, live time, accumulated dead time, accumulated live time)

As a consequence of this the file numbers MMM of the memory files do not need to be consecutive. In the above example memory 3 is using the file A01004.323.

To rename a bsl data set it is necessary to update the file names in the header file.

[+] In this case, DeltaTime and ExposureTime are currently not correctly updated

Version

V4.53 (2003-04-15)

Arguments

rapid2saxs [options] <i1nam> <onam> <odum> <i2mem> <i3mem> <bibo>

Options

-i1nam <rapid header file> : frame data

-i1mem <memory of frame data, default 1>

-i1fst <first frame, default 1>

-i1inc <increment, default 1>

-i1lst <number of frames>

-i2nam <scaler data file, default i1nam> : scaler data

-i2mem <memory of scaler data, default 2> (-i2nam /dev/null" to ignore scaler memory)

-i2fst <first frame, default 1>

-i2inc <increment, default 0>

-i3nam <timing data file, default i1nam> : timing data (-i3nam /dev/null" to ignore timing memory)

-i3mem <memory of timing data, default 3>

-i3fst <first frame, default 1>

-i3inc <increment, default 0>

Other options:

-bibo <bsl input byte order, default high byte first>: 1 or 2

-add <number of images to add>

sphere2saxs

Sphere2saxs can be used to generate a 2d scattering pattern. The scattering cross section can either be calculated for diluted spheres or a radial cross section profile can be read from a file. For mono disperse particles just give the radius in meter: -rad <radius[m]>. For a Schulz distribution write -rad "Schulz,<radius[m]>,<Z>". Z must be greater than -1. Alternatively, a radial scattering pattern can be read from a file with -rad "File,<filename>". Additional parameters can follow the filename, separated by commas, can be given to skip, swap and select columns. The default is that the first line contains the scattering vector length s in 1/nm (s=2 sin(Theta)/lambda) and the second line the specific scattering cross section (#scattered_photons/steradian)/#incoming_photons/thickness[m]. The default and only allowed reference system is Saxs. Detector rotation angles are taken into account. The output projection type is always Saxs (flat detector).

Version

2010-05-31 V4.67

Arguments

sphere2saxs [options] <onam> <odum> <odim 1> <odim 2> <opix 1> <opix 2>
<odis> <owvl> <rad> <dre> <vfr> <d><norm> <monv> <trm> <timv>

Options

dre [1, F]
Electron density diff. between particles and matrix [#/m^3] (default: 1e+28)

rad [1, s]
Particle radius [m] or distribution function (default: 1e-7)

Particle model:

- mono disperse : radius [m], e.g. sphere2saxs -rad 1e-7 ...
- Schulz distribution : "Schulz, radius [m], Z" with Z>-1, e.g. sphere2saxs -rad "Schulz,1e-7,1"
- from File : "File,name,skipl,skipc,swap,colx,coly", e.g. sphere2saxs -rad "File,xy.txt"

(defaults : name["xy.txt"], skipl[0], skipc[0], swap[1], colx[1], coly[2])
The parameter "File" reads a table that must contain in colx the scattering vectors s in 1/nm and in coly the differential scattering cross sections. All lines starting with # are automatically skipped. With option -d <thickness> the tabulated values are multiplied by <thickness>.

colx: s=2sin(Theta)/lambda [1/nm]

coly: 1/V*(dSigma/dOmega) or 1/A*(dSigma/dOmega) for sample thickness d=1

vfr [1, F]
Volume fraction (default: 0.01)

d [1, F]
Sample thickness [m] (default: 0.0001)

mca2saxs

Conversion of spec mca data to edf files (mca-frames, scalers). Only tested for spec files with one mca frame per scan

Version

V4.51 (2003-05-25)

Arguments

mca2saxs [options]
<i1nam> <onam> <i1fst> <i1lst> <i1inc>
<odum> <odim1> <odim2> <ofac> <ocon> <shft1> <shft2>

The scan number is chosen with i1fst.

Options

skip [1, D]
ignore the first skip scan numbers
(to access scans in the same file that have the same scan number)

Other Routines

Programs

col2array

A single column is extracted from a set of numbered ascii-input files and are rearranged to rows of a single ascii-output file. The columns must be separated by tabs ´\t´. The columns are numbered with 1, 2, 3 etc. from the start and -1, -2, -3 etc. from the end. The parameters must be entered in the described order. For omitted parameters default values are used.

Arguments

col2array [-h] [-b <mode[0]>] <output> <input> <first[1]> <last[1]> <inc[1]>
<col[1]> <skip_lines[0]> <skip_chars[0]> <separator['\t']>

- output : output filename, e.g. output.txt
- input : input filename with optional #-character specifying the position of
- the number, e.g. input_###.txt. Leading positions are filled with
- zeros.
- first : number of the first file (default: 1)
- last : number of the last file (default: first)
- inc : increment (default: 1)
- col : column number (default: 1)
- skip_lines : number of lines to skip at the top (default: 0)
- skip_chars : number of characters to skip (default: 0).

Options

-b <mode>: 0->text (default), 6->Signed32; 9->FloatIEEE32
-h : show this help

Example

The command

- `col2array output.txt input_###.txt 105 095 -3 -1 2 0`

will read the last column of the files input_105.txt, input_102.txt, input_099.txt and input_096.txt and write it to output.txt. It will skip 2 lines and 0 characters at the start of each file.

Macros

Macros can only be used if the path to the saxs programs is defined in the shell initialization script (e.g. ".cshrc"). The initialization script must not change the directory.

saxsdisp.mac

A single image of an edf file is converted into 8 bps tiff format and displayed with xv. The output image is automatically scaled between 0 and 255. Dummy values are set to 255. The macro may be aliased to xdisp.

Arguments

saxs_disp <input> [<num>]
input: input file name
num: image number (optional) (default: first image)
mem: memory number (optional) (default: memory 1)

Error Propagation

The error propagation is based on the following assumptions:
The errors of different data points are not correlated.
The error of each data point is described by its variance.
The variances are propagated using the laws of error propagation. Cross correlations are ignored.

The variance values are stored in a separate variance array. Usually, the file size is doubled when error propagation is switched on.

Mathematical Concept

Expectation Value, Variance and Covariance

The probability density f of a vector with n statistical variables x_i can be written as

(22) $$f(x_1, x_2, \cdots, x_n) \equiv f(\mathbf{x})$$

(23) $$\mathbf{E}[x_i] = \langle x_i \rangle = \int x_i \cdot f(x_1, x_2, \cdots, x_n) \cdot dx_1 dx_2 \cdots dx_n \equiv \int x_i \cdot f(\mathbf{x}) \cdot d\mathbf{x}$$

This can be written as the expectation value of an n-dimensional vector **x**:

(24) $$\mathbf{E}[\mathbf{x}] = (E[x_1], E[x_2], \cdots, E[x_n])$$

The variance is the expectation value E of the squares of the differences between **x** and its mean value E[**x**]:

(25) $$\mathbf{V} = \mathbf{V}[\mathbf{x}] = \mathbf{E}\left[(\mathbf{x} - \mathbf{E}[\mathbf{x}]) \cdot (\mathbf{x} - \mathbf{E}[\mathbf{x}])^T\right]$$

(26) $$\mathbf{V}[\mathbf{x}] = \begin{pmatrix} V_{11} & \cdots & V_{1n} \\ \vdots & Vij & \vdots \\ V_{n1} & \cdots & V_{nn} \end{pmatrix}$$

(27) $$V_{ii} = \mathrm{var}(x_i) = \int (x_i - \langle x_i \rangle)^2 \cdot f(\mathbf{x}) \cdot d\mathbf{x}$$

(28) $$V_{ij} = \mathrm{cov}(x_i, x_j) = \int (x_i - \langle x_i \rangle) \cdot (x_j - \langle x_j \rangle) \cdot f(\mathbf{x}) \cdot d\mathbf{x}$$

The matrix V[x] is called covariance matrix. It contains the variances V_{ii} and the covariances V_{ij} ($i \neq j$).

Principle of Error Propagation

The input variables are described with **x** and the output variables with **y**.

The variance array of the output array **V**[**y**] is calculated from the variance array of the input arrays **V**[**x**] using the derivative matrix **B**. The input array **x** has n variables (pixels), the output array **y** has m variables (pixels).

The error propagation is calculated by transforming the input covariance matrix **V**[**x**] to the output covariance matrix **V**[**y**] according to the rule of error propagation given in eq. (24):

$$\mathbf{V}[\mathbf{y}] = \mathbf{B} \cdot \mathbf{V}[\mathbf{x}] \cdot \mathbf{B}^T \tag{29}$$

The matrix **B** contains the derivatives of each output variable (output pixel) y_i with respect to each input variable x_j (input pixel):

$$B = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{pmatrix} \tag{30}$$

A straightforward application of this rule would result in an explosion of array sizes and calculation time. For practical purposes it must be simplified.

Error Propagation During 2d-data Reduction

Uncorrelated Input Data

In the case of uncorrelated input variables **x** the non-diagonal elements of the covariance matrix **V**[**x**] (eq. (21)) disappear and the diagonal elements $V[\mathbf{y}]_{ii}$ of the output covariance matrix **V**[**y**] can be calculated with :

$$V[\mathbf{y}]_{ii} = \sum_{k=1}^{n} \left(\frac{\partial y_i}{\partial x_k} \right)^2 \cdot V[x_k] \tag{31}$$

The non-diagonal elements of the output covariance matrix are

$$V[\mathbf{y}]_{ij} = \sum_{k=1}^{n} \left(\frac{\partial y_i}{\partial x_k} \right) \cdot \left(\frac{\partial y_j}{\partial x_k} \right) \cdot V[x_k] \tag{32}$$

If the B matrix is not diagonal the output variables are correlated.

Pixel by Pixel Calculations

Identical number of input and output variables

If the number n and m of the input and output variables are equal the derivative matrix B can be diagonal. In this case the component $V[\mathbf{y}]_{ij}$ of the output covariance matrix can be calculated directly from the component $V[\mathbf{x}]_{ij}$ of the input covariance matrix:

$$(33) \quad V[\mathbf{y}]_{ij} = \left(\frac{\partial y_i}{\partial x_i}\right) \cdot \left(\frac{\partial y_j}{\partial x_j}\right) \cdot V[\mathbf{x}]_{ij}$$

In this case the elements of the covariance matrix do not mix. The output variance array can be calculated independently for each variable, even if the input variables are correlated. This case corresponds to a pixel to pixel manipulation of a two dimensional data array, e.g. during scaling or normalization.

Smaller number of output variables than input variables

When L identically dimensioned input arrays with variables $\mathbf{x^1}$, $\mathbf{x^2}$, ... $_{\mathbf{x}}{}^{\mathbf{L}}$ are combined pixel by pixel to an output array with the same number of variables the derivative matrix B can be splitted into L diagonal subarrays of the form:

$$(34) \quad \mathbf{B^l} = \begin{pmatrix} \frac{\partial y_1}{\partial x_1^l} & 0 & & 0 \\ 0 & \frac{\partial y_2}{\partial x_2^l} & & 0 \\ & & \ddots & \\ 0 & 0 & & \frac{\partial y_n}{\partial x_n^l} \end{pmatrix}$$

The matrix B can be written as.

$$(35) \quad \mathbf{B} = \begin{pmatrix} \mathbf{B^1} & \mathbf{B^2} & \cdots & \mathbf{B^L} \end{pmatrix}$$

The covariance matrix V can be written in the same way, if any correlation between variables of different arrays can be excluded.

$$(36) \quad \mathbf{V} = \begin{pmatrix} \mathbf{V^1} & 0 & & 0 \\ 0 & \mathbf{V^2} & & 0 \\ & & \ddots & \\ 0 & 0 & & \mathbf{V^L} \end{pmatrix}$$

$$(37) \quad \mathbf{V}[\mathbf{y}] = \begin{pmatrix} \mathbf{B^1} & \mathbf{B^2} & \cdots & \mathbf{B^L} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{V^1} & 0 & & 0 \\ 0 & \mathbf{V^2} & & 0 \\ & & \ddots & \\ 0 & 0 & & \mathbf{V^L} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{B^1} \\ \mathbf{B^2} \\ \vdots \\ \mathbf{B^L} \end{pmatrix}$$

The component ij of the output covariance matrix is given by

$$(38) \quad V[\mathbf{y}]_{ij} = \sum_{l=1}^{L} \mathbf{B^l} \cdot \mathbf{V^l} \cdot \mathbf{B^l}\Big|_{ij} = \sum_{l=1}^{L} \left(\frac{\partial y_i}{\partial x_i^l}\right) \cdot \left(\frac{\partial y_j}{\partial x_j^l}\right) \cdot V[\mathbf{x^l}]_{ij}$$

The summation is running over all input arrays l

Pixel by pixel combination of L identical input data arrays with uncorrelated variables

In the case that all input data variables are uncorrelated also the output variables are uncorrelated. The variance of each output data point can be calculated with eq. (33). In this case it is only necessary to store additionally to each output data value its variance, which is the diagonal value of the covariance matrix.

$$(39) \quad V[\mathbf{y}]_{ii} = \sum_{l=1}^{L}\left(\frac{\partial y_i}{\partial x_i^l}\right)^2 \cdot V[\mathbf{x}^l]_{ii}$$

where $V[\mathbf{y}]_{ii} = V[y_i] = \sigma^2_{yi}$ is the variance of data point i of the output array and $V[\mathbf{x}^l]_{ii} = V[x^l_i] = \sigma^2_{xli}$ the variance of the i-th data point of the l-th input data array.

Correlations between data points are not taken into account. This assumption is strictly valid if the calculations are done pixel by pixel. It is also valid if images are binned. It is generally not strictly valid if pixels are artificially divided into sub-pixels. This happens if an image is mapped to an image with different binning size. It leads to spatial smearing of the image.

Explicit Error Propagation Formulas

Remapping of data arrays

This operation writes the pixel intensities of region[i,j] of input array In1 into pixel Out[i,j] of the output array. A pixel has the area 1. The value of a partial pixel is calculated by multiplying the value of the full pixel with the partial pixel area.
A region is a rectangular area of an array. The sum of a region is the sum of all partial pixel values inside the region, the weight is the sum of all partial pixel areas inside the region. Dummy pixels are not take into account.

Sum(In1,region[i,j]): sum of all non-dummy partial pixel values in region[i,j] of array In1

Weight(In1, region[i,j]): sum of all non-dummy partial pixel areas in region[i,j] of array In1

a) Summation of data values

Out[i,j] = Sum(In1, region [i,j])

VOut[i,j] = Sum(VIn1, region [i,j])

b) Averaging of data values

Out[i,j] = Sum(In1, region [i,j])/Weight(In1, region [i,j])

VOut[i,j] = Sum(VIn1, region [i,j])/Weight(In1, region [i,j])^2

Scaling of two data arrays

Out[i,j] = In1[i,j] * factor1 + constant1

VOut[i,j] = VIn1[i,j] * factor1^2

Summation of two data arrays

Out[i,j] = In1[i,j] + In2[i,j]

VOut[i,j] = VIn1[i,j] + VIn2[i,j]

Multiplication of two data arrays

Out[i,j] = In1[i,j] * In2[i,j]

VOut[i,j] = VIn1[i,j] * In2[i,j] ^2 + In1[i,j] ^2 * VIn2[i,j]

Division of two data arrays

Out[i,j] = In1[i,j] / In2[i,j]

VOut[i,j] = (VIn1[i,j] + Out[i,j]^2 * VIn2[i,j]) / In2[i,j]^2

Implementation of Error Propagation

The error propagation calculation is based on the propagation of variances pixel by pixel following the rules for error propagation. Cross correlations are not taken into account. During binning and changes of pixel sizes the variances are transformed accordingly

Usage

To use error propagation, the variance of the input image(s) needs to be defined. The variance calculations are done automatically if a variance array exists for the first input image. Variances are written if the variance of the output image is defined (either with -oerr <expression> or by error propagation from the input image(s). The default variance of an input image is 0 (exact).

Error propagation options for all saxs-programs:

-iNerr <expression(_iNval)>|
<expression(_oval)> : define variance of input/output image,
e.g. -i1err "_i1val+pow(1.6,2)" : set the variance of the input image to the input pixel value (Poisson, assuming 1 ADU/pixel = 1 detected photon/pixel), and add static noise with a standard deviation of sigma=1.6.
+/-var use/do not use variance arrays

The file size is usually doubled when variance arrays are used. If a file contains image data and variance data, saxs_ascii creates two output files, the first contains the usual data, the second is written in the same format as the first one and contains the standard deviations (square root of the variances). The file name of the second file ends with stddev.txt.

Access

If several images with variance arrays are written into the same EDF file, the image blocks can be displayed with the program "onze". The variance blocks cannot be displayed. Fit2d displays only the first image block. It must be an image block.

The following command extracts variance blocks from file.edf and writes them as images to file_var.edf:

- `saxs_mac -i1mem -1 -omem 1 file.edf file_var.edf`

The images in file_var.edf can be displayed with onze. The first one in the file can also be displayed with fit2d.

The macro saxs_disp (saxs_disp <input> [<num> [<mem>]]) uses xv and allows to display data blocks (mem=1) and variance blocks (mem=-1).

Format

The file format follows the specification of EDF_DataFormatVersion=2.40 as described in SaxsKeywords.pdf (2004-12-05). The variance block has the data block identifier (DataBlockID) N.Image.Error, where N is the so-called "Image" number of the data block. The data block identifier of the variance block to image number 1 is 1.Image.Error. For conveniencce, the variance block is written after the image data block and the image number is repeated after the keyword "Image.Error". This keyword is ignored.

The dimensions of the variance array must be exactly the same as for the image array. If they differ, the variances are ignored or an error message is given.

History

2005-02-27	Continuation lines for history lines
2005-09-20	Error propagation included into all saxs-programs
2006-10-17	+-aa, history macros, isotime programs

Index