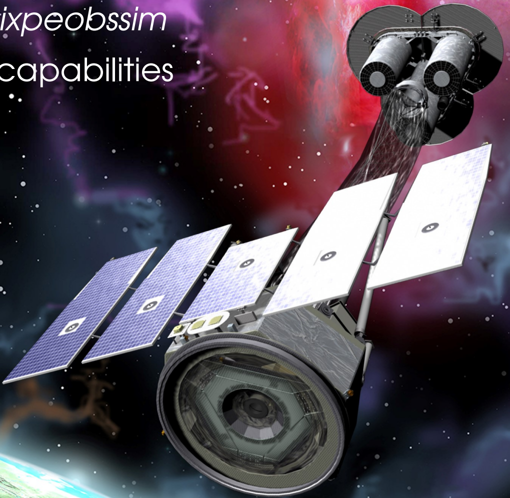
# Introduction to *ixpeobssim* & its simulation capabilities



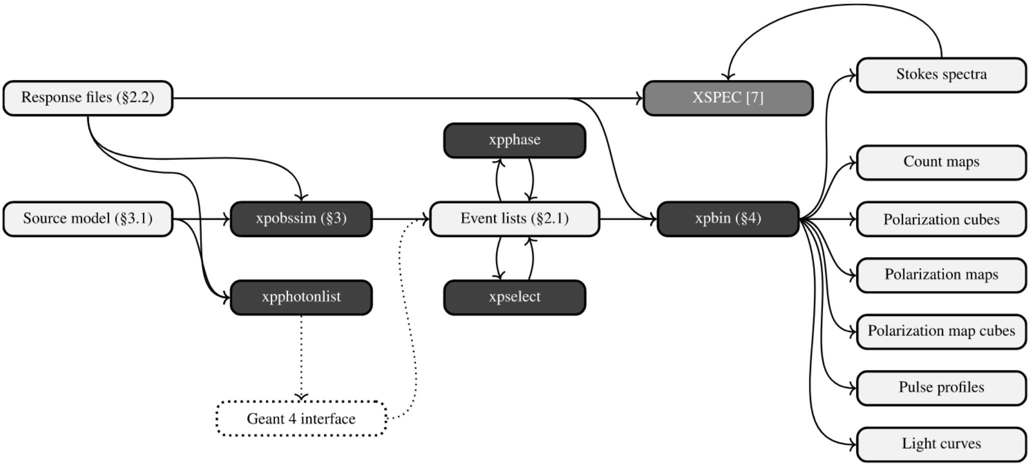
Niccolò Di Lalla
niccolo.dilalla@stanford.edu
Stanford University

Joint NICER/IXPE Workshop 2024

# Introduction

- Project started in 2015 under the name of `XIMPOL`:
  - Initially not tied to any specific mission or instrument design
  - After IXPE selection in 2017, it was renamed and progressively tailored in preparation for the new mission
  - Publicly released in 2022 to support the analysis of public IXPE data and engage the broader community in anticipation of the General Observer program
- Simulation and analysis framework:
  - Based on `python` programming language and the associated scientific ecosystem
  - Designed to produce fast and realistic simulated IXPE observations
  - Complemented by a suite of post-processing applications to select, bin and analyze simulated and real IXPE data
  - Modular nature allows for the implementation of complex, polarization-aware analysis pipelines
- Output data are:
  - Event lists in FITS format, containing a strict superset of the information included in the publicly released IXPE data products
  - Fully compliant with the visualization and analysis tools commonly used by the X-ray community (`XSPEC`, `Sherpa`, `3ML`, `DS9`, `HENDRICS`).

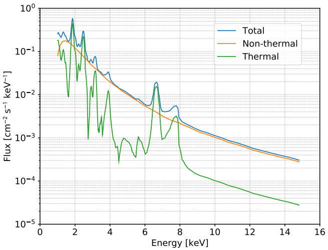
# Architectural Overview

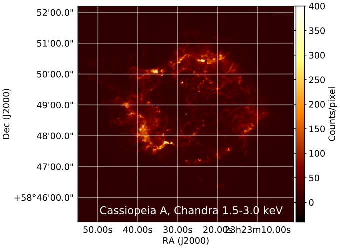


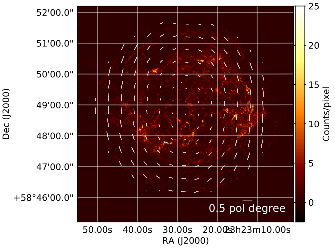
# Source Model Definition

https://ixpeobssim.readthedocs.io/en/latest/source_models.html

- Need to define essentially three source properties:
  - Morphology (point sources, disks, annuli, generic extended sources from FITS images)
  - Energy spectrum in units of $[\mathrm{cm}^{-1}\,\mathrm{s}^{-1}\,\mathrm{keV}^{-1}]$ (arbitrary Python function of energy and time or an XSPEC model)
  - Polarization model (degree and angle, or Stokes parameters Q and U)
- Can use a Chandra photon list in lieu of defining morphology and spectrum
- Can overlay several components or sources in the same ROI
- Support for time-dependent transient and periodic sources



# Configuration File: Point Source with Background

https://drive.google.com/file/d/1fUumSqttW8wDK5mGCo4JpTUIGUxrwJkC/view?usp=sharing

```
# Source and pointing coordinates, in decimal degrees
SRC_NAME = 'Toy point source w/ bkg'
SRC_RA, SRC_DEC = 45., 45.
PNT_RA, PNT_DEC = SRC_RA, SRC_DEC

# Spectral and polarimetric parameters
PL_NORM = 1          # cm-1 s-1 keV-1 @ 1 keV
PL_INDEX = 2.        # -2
PA = 30.             # 30 degrees
PD_INT = 0.1         # 10% @ 1 keV
PD_SLOPE = 0.05      # +5% / keV

# Definition of the spectral and polarization models
spec = power_law(PL_NORM, PL_INDEX)
pol_ang = constant(numpy.radians(PA))
def pol_deg(E, t=None, ra=None, dec=None):
    """Linear model for the polarization degree vs. energy.
    """
    return numpy.clip(PD_INT + PD_SLOPE * (E - 1.), 0., 1.)

# Definition of the sources and the region of interest.
src = xPointSource(SRC_NAME, SRC_RA, SRC_DEC, spec, pol_deg, pol_ang)
bkg = xTemplateInstrumentalBkg()
ROI_MODEL = xROIModel(PNT_RA, PNT_DEC, src, bkg)
```

# Configuration File: Toy Cas A (1/2)

https://github.com/lucabaldini/ixpeobssim/blob/main/ixpeobssim/config/toy_casa.py

```python
# Coordinates of the pointing.
RA_PNT = 350.8664167
DEC_PNT = 58.8117778
# Maximum polarization degree, and corresponding radius.
MAX_POL_DEG = 0.5
MAX_RADIUS = arcmin_to_degrees(2.8)

def _load_spec(file_name, emin=1., emax=15.):
    """Convenience function to load a spectral csv file.
    """
    file_path = os.path.join(IXPEOBSSIM_CONFIG_ASCII, file_name)
    return load_spectral_spline(file_path, emin=emin, emax=emax, delimiter=',')

# Read in the spectral models from the proper files.
total_spec_spline = _load_spec('casa_total_spectrum.csv')
non_therm_spec_spline = _load_spec('casa_nonthermal_spectrum.csv')
therm_spec_spline = total_spec_spline - non_therm_spec_spline

# Define the spectral models.
def therm_spec(E, t):
    return therm_spec_spline(E)

def non_therm_spec(E, t):
    return non_therm_spec_spline(E)
```

# Configuration File: Toy Cas A (2/2)

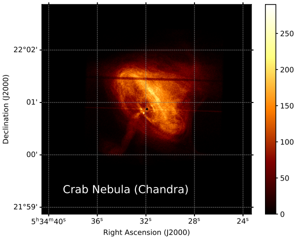
https://github.com/lucabaldini/ixpeobssim/blob/main/ixpeobssim/config/toy_casa.py

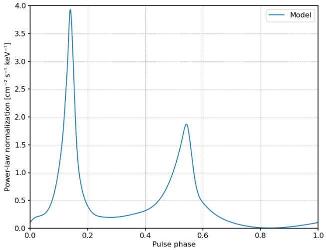
```
# The thermal component is unpolarized.
therm_pol_ang = constant(0.)
therm_pol_deg = constant(0.)

# The non thermal component is tangentially polarized.
def non_therm_radial_profile(r, E=None, t=None):
    return numpy.clip((MAX_POL_DEG * r / MAX_RADIUS), 0., 1.)
non_therm_pol_field = xTangentialPolarizationField(RA_PNT, DEC_PNT, non_therm_radial_profile)
non_therm_pol_deg = non_therm_pol_field.polarization_degree_model()
non_therm_pol_ang = non_therm_pol_field.polarization_angle_model()

# Finally, the Chandra images for the thermal and non thermal components.
le_img_file_path = os.path.join(IXPEOBSSIM_CONFIG_FITS, 'casa_1p5_3p0_keV.fits')
he_img_file_path = os.path.join(IXPEOBSSIM_CONFIG_FITS, 'casa_4p0_6p0_keV.fits')

# Create the model components.
therm_comp = xExtendedSource('Cas A thermal', le_img_file_path, therm_spec,
    therm_pol_deg, therm_pol_ang)
non_therm_comp = xExtendedSource('Cas A non-thermal', he_img_file_path, non_therm_spec,
    non_therm_pol_deg, non_therm_pol_ang)
instrumental_bkg = xTemplateInstrumentalBkg()

# Create the actual ROI object.
ROI_MODEL = xROIModel(RA_PNT, DEC_PNT)
ROI_MODEL.add_sources(therm_comp, non_therm_comp, instrumental_bkg)
```

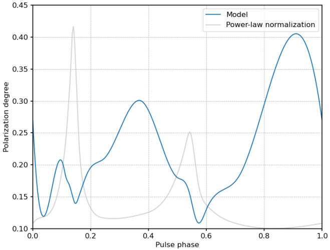
# Toy Crab Complex Model

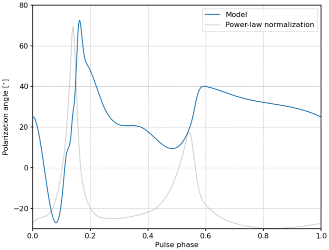
https://drive.google.com/file/d/1Aa5_vgqo1LkL_66Jx8yDXUuaBmiUmyPp/view

- Chandra observation is used to define the ROI
- Given its brightness, the region of the Crab Pulsar suffers of strong pile-up and is removed from the simulation.
- A new periodic point source is added at the pulsar location:
    - Spectral and polarization models as a function of the pulse phase.



# Configuration File: Toy Crab Complex (1/2)

https://drive.google.com/file/d/1Aa5_vgqo1LkL_66Jx8yDXUuaBmiUmyPp/view?usp=sharing

```python
# Parse the phase-resolved pulsar parameters.
phase, norm, pd, pa = numpy.loadtxt(file_path='crab_pulsar2.txt', unpack=True)
pa = numpy.deg2rad(pa - 90.)

# Build the spline for the PL normalization and the phase-resolved energy spectral model.
pl_norm_spline = xInterpolatedUnivariateSpline(phase, norm, k=3)
spec = power_law(pl_norm_spline, pl_index=1.6)

# Build the spline for the polarization degree and angle and the corresponding models.
pol_deg_spline = xInterpolatedUnivariateSpline(phase, pd, k=3)
def pol_deg(E, phase, ra=None, dec=None):
    return pol_deg_spline(phase)

pol_ang_spline = xInterpolatedUnivariateSpline(phase, pa, k=3)
def pol_ang(E, phase, ra=None, dec=None):
    return pol_ang_spline(phase)

# Define the Crab Pulsar.
PSRcoord = SkyCoord('05h34m31.93830s', '22d00m52.1758s', frame='icrs')
ephemeris = xEphemeris(mjd_to_met(59380), nu0=29.5948563919, nudot0=-3.6770137e-10,
    nuddot=1.18e-20)
crab_psr = xPeriodicPointSource('Crab_pulsar', PSRcoord.ra.deg, PSRcoord.dec.deg, spec,
    pol_deg, pol_ang, ephemeris)
```

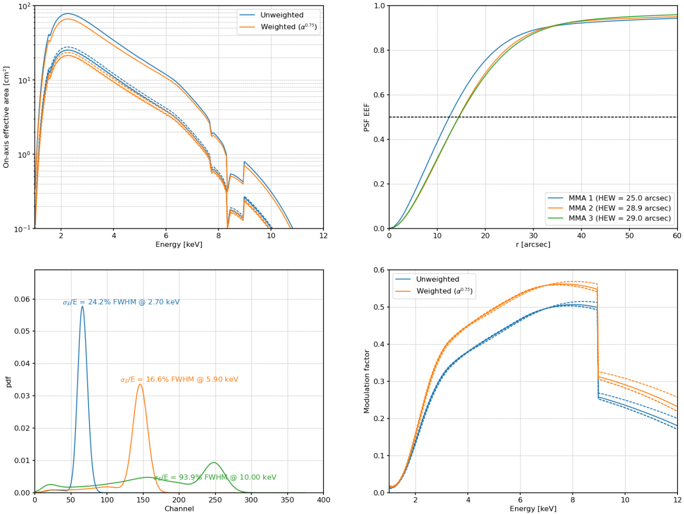
# Configuration File: Toy Crab Complex (2/2)

https://drive.google.com/file/d/1Aa5_vgqo1LkL_66Jx8yDXUuaBmiUmyPp/view?usp=sharing

```
# Build the ROI model.
ROI_MODEL = xChandraROIModel('acisf16364_repro_evt2.fits', acis='S')

# Remove the pulsar region and add our pulsar model.
region = CircleSkyRegion(center=PSRcoord, radius=1.5 * u.arcsec)
remove_psr = xChandraObservation('Remove', None, None, region, exclude=True)
ROI_MODEL.add_source(remove_psr)
ROI_MODEL.add_source(crab_psr)

# Build a tangential polarization profile for the Crab Nebula.
def pol_radial_profile(r, E=None, t=None):
    return numpy.clip((0.5 * r / arcmin_to_degrees(1.5)), 0., 1.)
pwn_pol_field = xTangentialPolarizationField(ra_psr, dec_psr, pol_radial_profile)
pwn_pol_deg = pwn_pol_field.polarization_degree_model()
pwn_pol_ang = pwn_pol_field.polarization_angle_model()

# Create the Crab Nebula model from Chandra and add it to the ROI.
crab_pwn = xChandraObservation('Crab_PWN', pwn_pol_deg, pwn_pol_ang)
ROI_MODEL.add_source(crab_pwn)

# Finally, add the instrumental background.
instrumental_bkg = xTemplateInstrumentalBkg()
ROI_MODEL.add_source(instrumental_bkg)
```

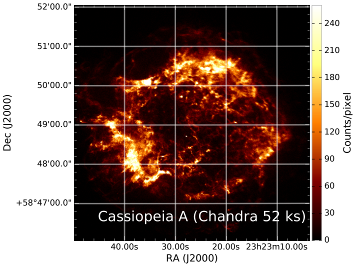
# Instrument Response Functions

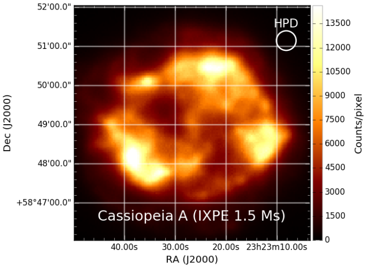
- Each of the three DUs has its own set of IRFs:
  - FITS files compliant with the OGIP format
  - Weighted and un-weighted flavors
- Generated and stored in a local CALDB:
  - Kept in sync with the official IXPE CALDB
- Latest version (v13) released in March 2024:
  - Time-dependent, validity time binned in 6-month interval
  - User has to select the appropriate set of IRFs

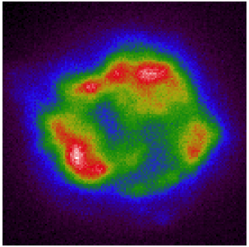


# *xpobssim*

## Simulation flow



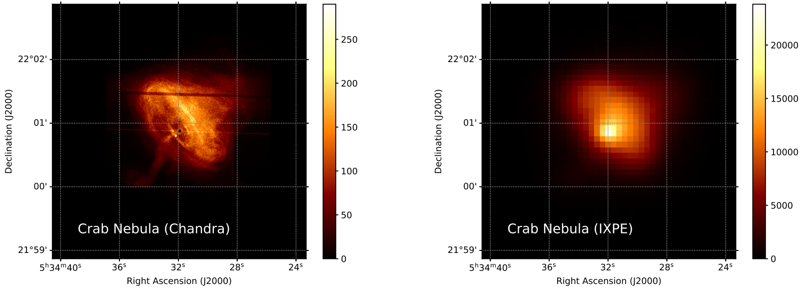



- Simulate an observation starting from an arbitrary, user-provided source model:
  - Calculate the expected number of events by convolving the source spectrum with the effective area and extract the event times based on the light curve
  - Extract the true energies and sky positions and smear them with energy dispersion and PSF
  - Generate the angular distribution of the photoelectrons according to the polarization model
- With composite sources, the simulation is performed separately for each component and the resulting photon lists are then merged before applying the dead time effect

# Conversion of a Chandra Observation

## Simulation flow




- Process an actual archived Chandra photon list to produce an IXPE observation simulation:
  - Chandra measured energies, times and positions taken as MC truth
  - Events are down-sampled and smeared according to the IXPE response functions
  - The angular distribution of the photoelectrons is generated according to the provided polarization model
- Preserve the full correlation between the morphology and the energy spectrum

# How to run *xpobssim*

https://ixpeobssim.readthedocs.io/en/latest/reference.html

| Index | Extension | Type | Dimension | View | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | Primary | Image | 0 | Header | Image | Table | | |
| 1 | EVENTS | Binary | 20 cols X 2648312 rows | Header | Hist | Plot | All | Select |
| 2 | MONTE_CARLO | Binary | 9 cols X 2648312 rows | Header | Hist | Plot | All | Select |
| 3 | GTI | Binary | 2 cols X 1 rows | Header | Hist | Plot | All | Select |
| 4 | ROITABLE | Binary | 2 cols X 3 rows | Header | Hist | Plot | All | Select |
| 5 | TIMELINE | Binary | 4 cols X 1 rows | Header | Hist | Plot | All | Select |
| 6 | SC_DATA | Binary | 9 cols X 8002 rows | Header | Hist | Plot | All | Select |

- *xpobssim* (and all the other applications in *ixpeobssim*) can be either used as a stand-alone tool run as a shell command or easily combined into complex simulation and analysis scripts in `python`:
  - Fully configurable via command-line options or keyword arguments (seed, IRFs, SAA, vignetting, dithering, roll angle, occultation...)
  - Python wrappers return the list of all the files created which makes it easy to chain applications one after the other

```
xpobssim.py --duration=100000 --configfile=config/toy_point_source_bkg.py
```

```python
import ixpeobssim.core.pipeline as pipeline

file_list = pipeline.xpobssim(duration=duration, configfile='toy_point_source_bkg.py')
```

# Analysis Tools

- *ixpeobssim* is distributed with its own set of analysis tools:
  - Provide an easy-to-use interface to manipulate simulated and real IXPE data
  - Full support for weighted and un-weighted types of analysis
- *xpphase*:
  - Calculate the phase of a periodic source based on its ephemeris
- *xpselect*:
  - Filter event lists based on energy, direction, time or phase
- *xpbin*:
  - Bin the data using several different algorithms, producing binned events lists
  - HEASOFT `xselect` FTOOL provides support for part of the same functionalities since v6.30
- *xpbinview*:
  - Visualize the binned data products
- *xpstokesalign*:
  - Align the Stokes parameters to a given polarization model on an event-by-event basis
- *xpxspec*:
  - Perform a spectro-polarimetric fit in XSPEC

# References

- Github webpage: https://github.com/lucabaldini/ixpeobssim
- Software documentation: https://ixpeobssim.readthedocs.io/en/latest/index.html
- Paper: https://www.sciencedirect.com/science/article/pii/S2352711022001169
- Pip: https://pypi.org/project/ixpeobssim/
  - pip install ixpeobssim
  - Latest version: 31.0.1 released on March 8, 2024
- Tutorial @HEAD20: https://drive.google.com/drive/folders/1AGixwB3TSLGvMeQ89ICE-Ww6FL2QSXbe?usp=sharing

SoftwareX 19 (2022) 101194

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

*ixpeobssim*: A simulation and analysis framework for the imaging X-ray polarimetry explorer

Luca Baldini [a,b,*], Niccolò Bucciantini [c,d,e], Niccolò Di Lalla [f], Steven Ehlert [g], Alberto Manfreda [b], Michela Negro [h,i,j], Nicola Omodei [f], Melissa Pesce-Rollins [b], Carmelo Sgrò [b], Stefano Silvestri [a,b]

[a] Università di Pisa, Dipartimento di Fisica Enrico Fermi, Largo B. Pontecorvo 3, I-56127 Pisa, Italy
[b] Istituto Nazionale di Fisica Nucleare, Sezione di Pisa, Largo B. Pontecorvo 3, I-56127 Pisa, Italy
[c] Istituto Nazionale di Astrofisica, Osservatorio Astrofisico di Arcetri, Largo E. Fermi 5, I-50125, Firenze, Italy
[d] Dipartimento di Fisica & Astronomia, Univesità degli Studi di Firenze, Via G. Sansone 1, 50019, Sesto F.no, Italy
[e] Istituto Nazionale di Fisica Nucleare, Sezione di Firenze, Via G. Sansone 1, 50019 Sesto F.no, Italy
[f] W.W. Hansen Experimental Physics Laboratory, Kavli Institute for Particle Astrophysics and Cosmology, Department of Physics and SLAC National Accelerator Laboratory, Stanford University, Stanford, CA 94305, USA
[g] NASA Marshall Space Flight Center, Huntsville, AL 35812, USA
[h] University of Maryland, Baltimore County, Baltimore, MD 21250, USA
[i] NASA Goddard Space Flight Center, Greenbelt, MD 20771, USA
[j] Center for Research and Exploration in Space Science and Technology, NASA/GSFC, Greenbelt, MD 20771, USA

ARTICLE INFO

*Article history:*
Received 12 March 2022
Received in revised form 21 July 2022
Accepted 20 August 2022

*Keywords:*
X-ray polarimetry

ABSTRACT

*ixpeobssim* is a simulation and analysis framework specifically developed for the Imaging X-ray Polarimetry Explorer (IXPE). Given a source model and the response functions of the telescopes, it is designed to produce realistic simulated observations, in the form of event lists in FITS format, containing a strict superset of the information included in the publicly released IXPE data products. The core simulation capabilities are complemented by a full suite of post-processing applications which support the spatial, spectral, and temporal models needed for analysis of typical polarized X-ray sources, allowing for the implementation of complex, polarization-aware analysis pipelines, and facilitating the interoperation with the standard visualization and analysis tools traditionally in use by the X-ray community. Although much of the framework is specific to IXPE, the modular nature of the underlying implementation makes it potentially straightforward to adapt it to different missions with polarization capabilities.



Code metadata

| | |
|---|---|
| Current code version | 29.0.0 |
| Permanent link to code/repository used for this code version | https://github.com/ElsevierSoftwareX/SOFTX-D-22-00065 |
| Permanent link to Reproducible Capsule | |
| Legal Code License | GPL-3.0 |
| Code versioning system used | git |
| Software code languages, tools, and services used | python |
| Compilation requirements, operating environments & dependencies | numpy, scipy, matplotlib, astropy, regions, skyfield |
| If available Link to developer documentation/manual | https://ixpeobssim.readthedocs.io |
| Support email for questions | luca.baldini@pi.infn.it |

## 1. Introduction

Launched on December 9, 2021, the Imaging X-ray Polarimetry Explorer (IXPE) is a NASA Small Explorer Mission developed in collaboration with the Italian Space Agency [1–3], and the first

* Corresponding author at: Università di Pisa, Dipartimento di Fisica Enrico Fermi, Largo B. Pontecorvo 3, I-56127 Pisa, Italy.
*E-mail address:* luca.baldini@pi.infn.it (Luca Baldini).

https://doi.org/10.1016/j.softx.2022.101194

# Conclusions

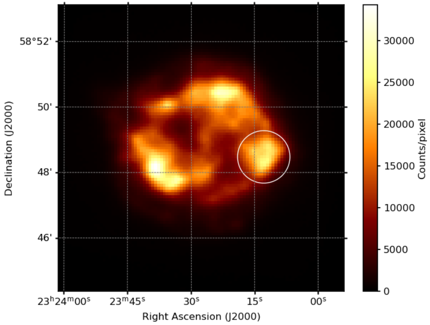
- *ixpeobssim* was developed to support the IXPE mission by providing advanced simulation and analysis facilities
- With the official public release of IXPE data, we decided to release the codebase under an OSI-approved license:
  - Support the community engaged in the analysis of IXPE data
  - Support the simulation effort required for the General Observer program
  - Encourage reuse for future X-ray (polarimetry?) missions
- *ixpeobssim* is stable but still under an active development phase:
  - A new release every few months (check the release notes!)
  - Please open a new issue on github if you find a bug or have something to propose/discuss
  - Everyone is very welcome to participate and help us with the development!
- Many areas can be improved and are currently in the works:
  - Improve the current simplistic, azimuthally-symmetric model for the PSF
  - Implement a tool to quickly evaluate the effect of the polarization leakage
  - Improve the plotting routines
  - Add new analysis tools and algorithms

# SPARE SLIDES

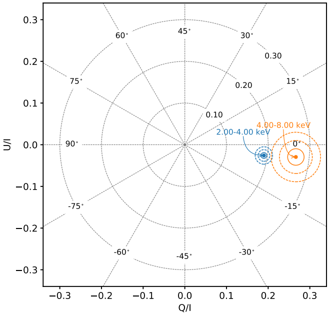
# Polarization cubes

## Binned Data Products

- The simplest possible data structure holding polarization information
- Table listing I, Q, U, polarization degree and angle with the associated uncertainties in multiple energy bins:
  - Provided with methods to rescale and subtract the background contribution (see talk by Stefano)



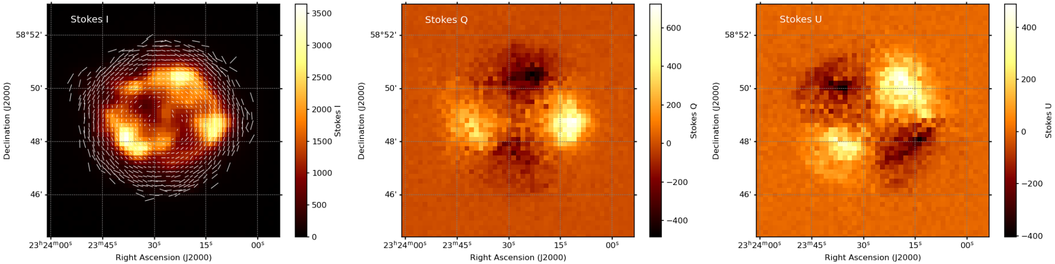

→



# Polarization maps

## Binned Data Products

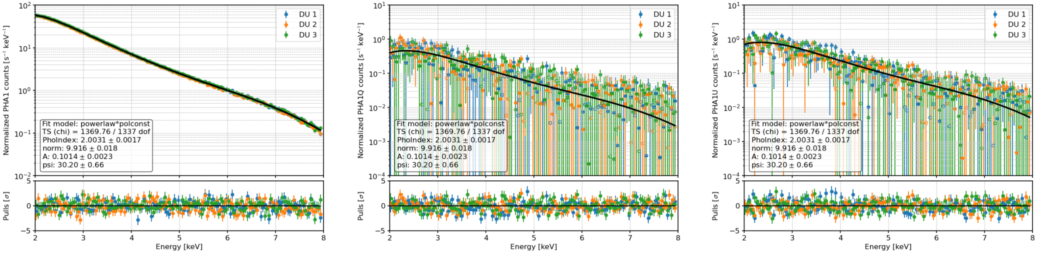
- Hold the exact same information as polarization cubes, but binned in sky-coordinates
- Maps of I, Q, U, polarization degree and angle in multiple energy layers:
  - Provided with methods to convolve the map with a generic binned kernel and overlay the arrows of polarization information



# Stokes spectra

## Binned Data Products

- Main interface to spectro-polarimetric fitting in `XSPEC`, `Sherpa` and `3ML`
- Generalization of the standard PHA spectra:
  - PHA1, PHA1Q and PHA1U
- Lightweight python wrapper dubbed *xpxspec* shipped with *ixpeobssim*:
  - Together with a few simple, multiplicative polarimetric models provided by HEASARC through the page hosting XSPEC additional models



# Simulation and Analysis

## Point source with background example

```
def simulate(duration=100000):
    """Run the simulation.
    """
    pipeline.xpobssim(duration=duration, configfile='toy_point_source_bkg.py')

def select(src_rad=0.75, bkg_inner_rad=1.5, bkg_outer_rad=3.):
    """Select the photon lists.
    """
    file_list = pipeline.file_list()
    pipeline.xpselect(*file_list, rad=src_rad, suffix='src')
    pipeline.xpselect(*file_list, innerrad=bkg_inner_rad, rad=bkg_outer_rad, suffix='bkg')

def bin_(ebinning=[2, 4, 8]):
    """Create the necessary binned files.
    """
    pipeline.xpbin(*pipeline.file_list(), algorithm='CMAP')
    kwargs = dict(algorithm='PCUBE', ebinalg='LIST', ebinning=ebinning)
    pipeline.xpbin(*pipeline.file_list('src'), **kwargs)
    pipeline.xpbin(*pipeline.file_list('bkg'), **kwargs)
    for algorithm in ['PHA1', 'PHA1Q', 'PHA1U']:
        pipeline.xpbin(*pipeline.file_list('src'), algorithm=algorithm)
        pipeline.xpbin(*pipeline.file_list('bkg'), algorithm=algorithm)
```

# Basic polarization analysis using Stokes parameters

Kislat et al., Astroparticle Physics **68**, 45–51 (2015)

- For each measured angle $\phi_k$, a set of Stokes parameters can be defined as:

$$i_k = 1,$$
$$q_k = \cos 2\phi_k$$
$$u_k = \sin 2\phi_k.$$

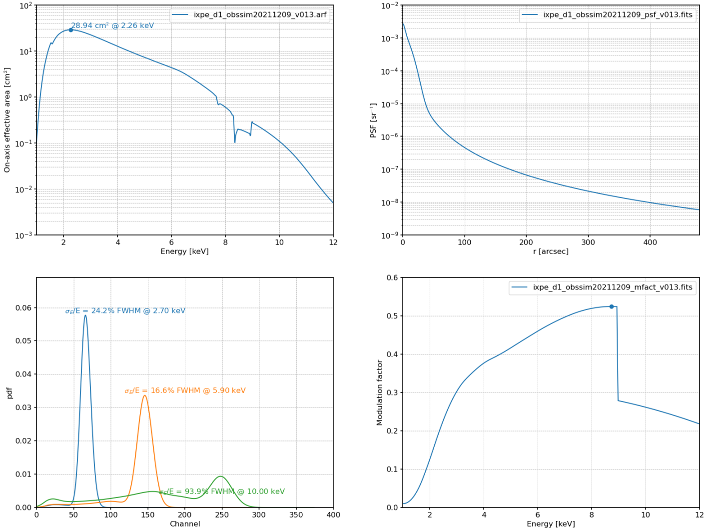
- Owing to their linearity, the analysis for a data-set consisting of $N$ events reduces to:

$$I = \sum_{k=1}^{N} i_k = N, \qquad Q = \sum_{k=1}^{N} q_k, \qquad U = \sum_{k=1}^{N} u_k.$$

- Finally, the degree and angle of polarization can be estimated as:

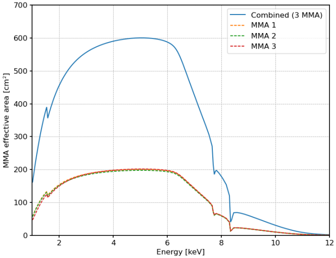
$$P = \frac{2}{\mu}\frac{\sqrt{Q^2 + U^2}}{I}$$
$$\phi = \frac{1}{2}\arctan\frac{U}{Q}.$$

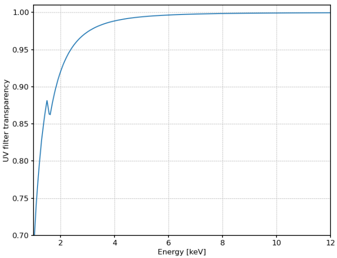
# Instrument Response Functions

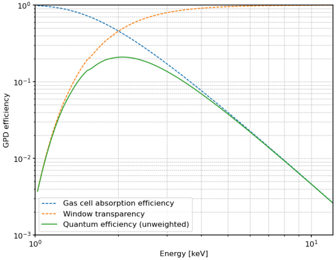


# IXPE effective area

## Relevant contributions



# IXPE field of view