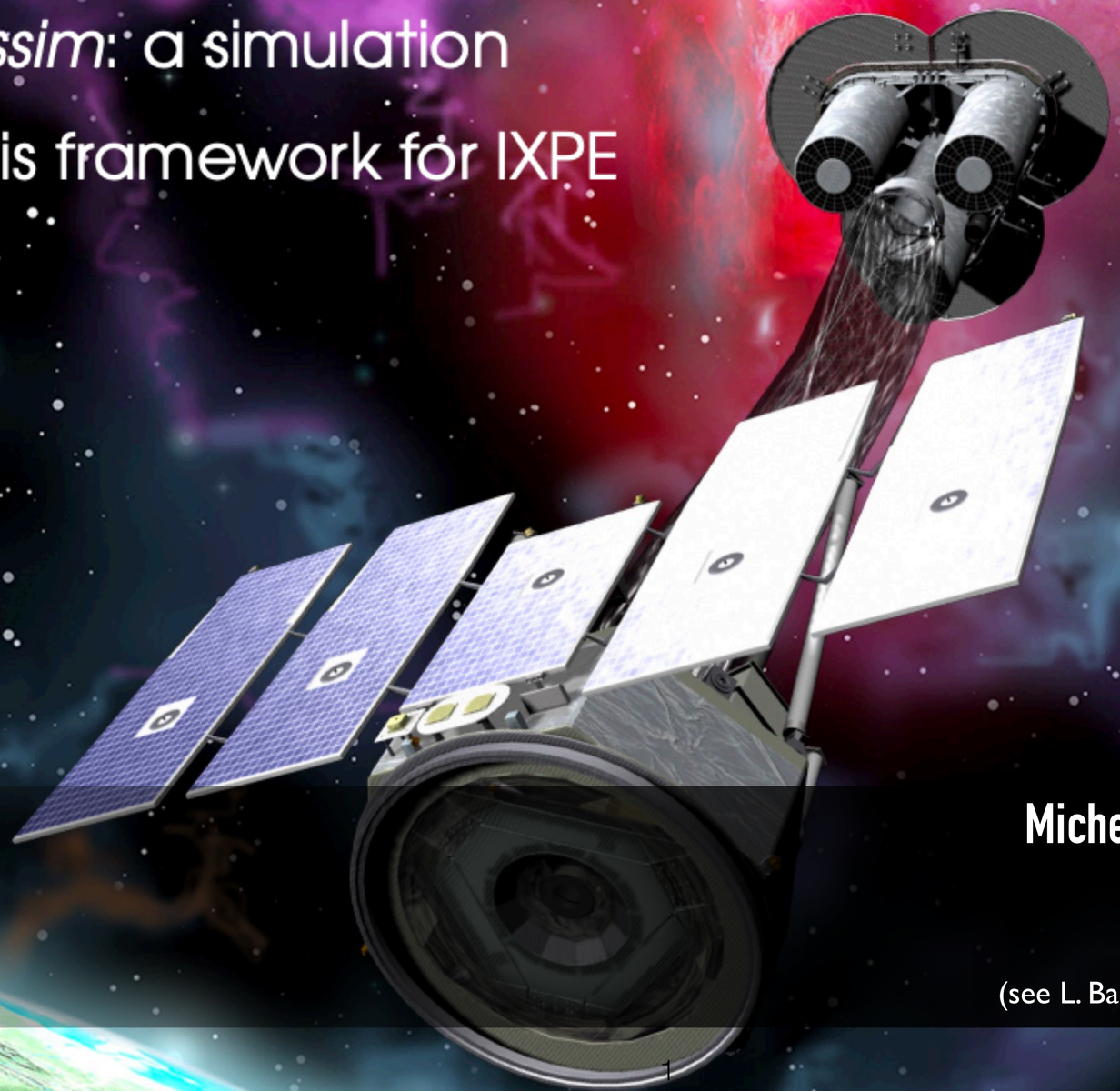
ixpeobssim: a simulation and analysis framework for IXPE



Michela Negro — Louisiana State University
Niccolò Di Lalla — Stanford University

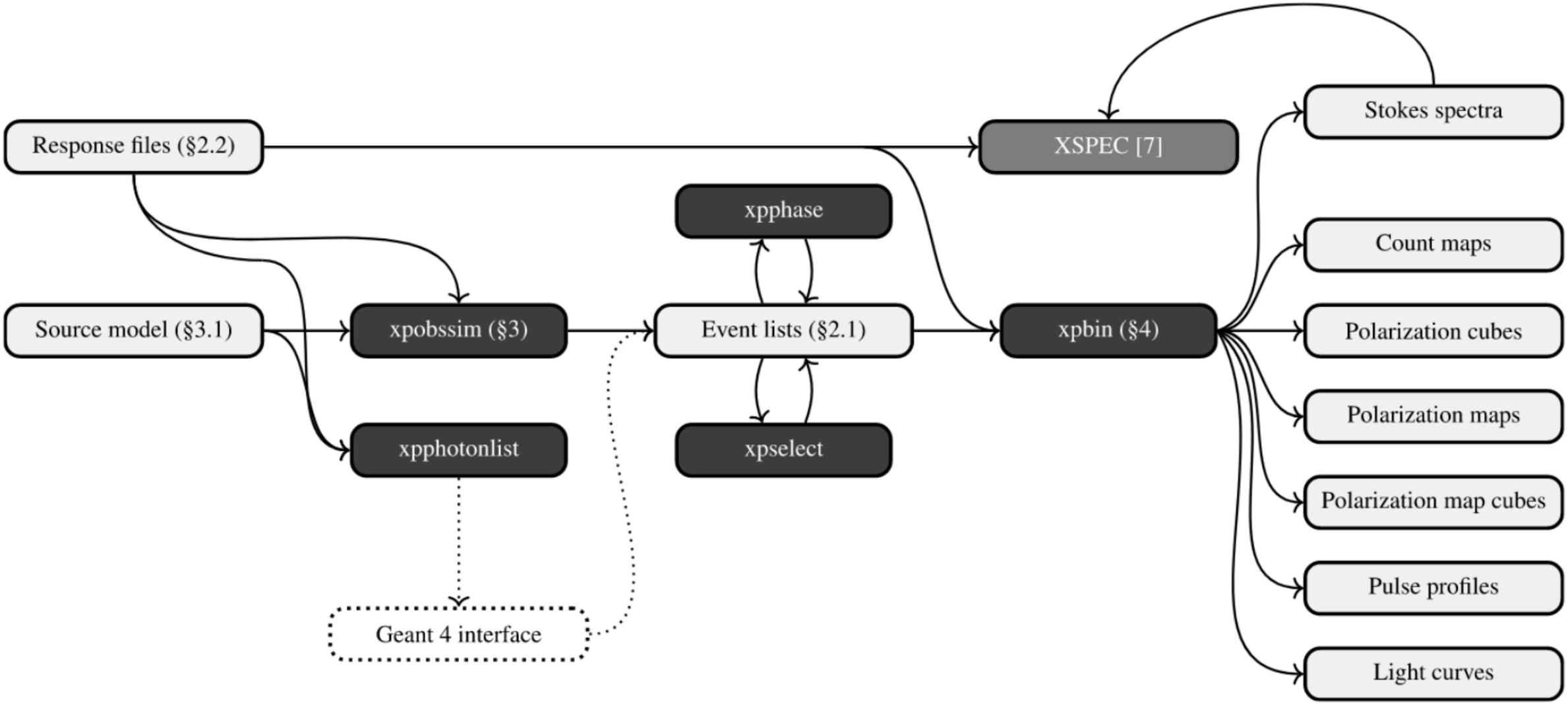
Based on work done by a larger group
(see L. Baldidni et al., 2022, SoftwareX, Volume 19, id. 101194)

Simulation and Analysis

- 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

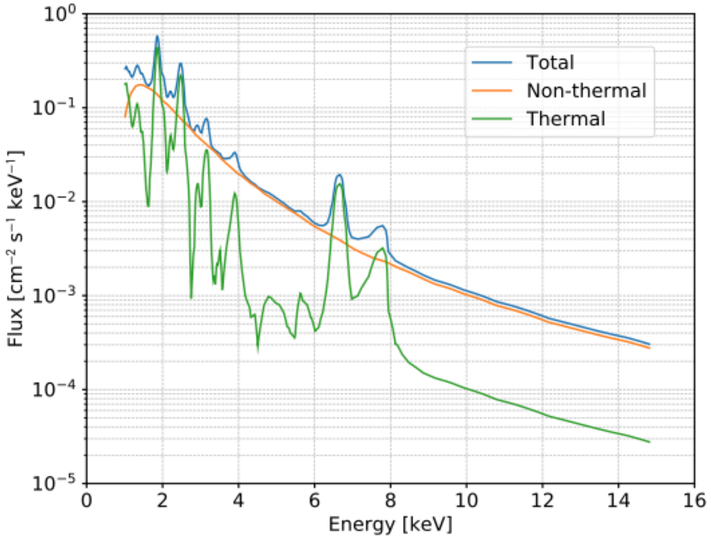
- Output data are:
  - Event lists in FITS format, containing a strict superset of the information included in the publicly released IXPE data products
  - Fully compliant with the visualization and analysis tools commonly used by the X-ray community (`XSPEC`, `Sherpa`, `3ML`, `DS9`, `HENDRICS`).

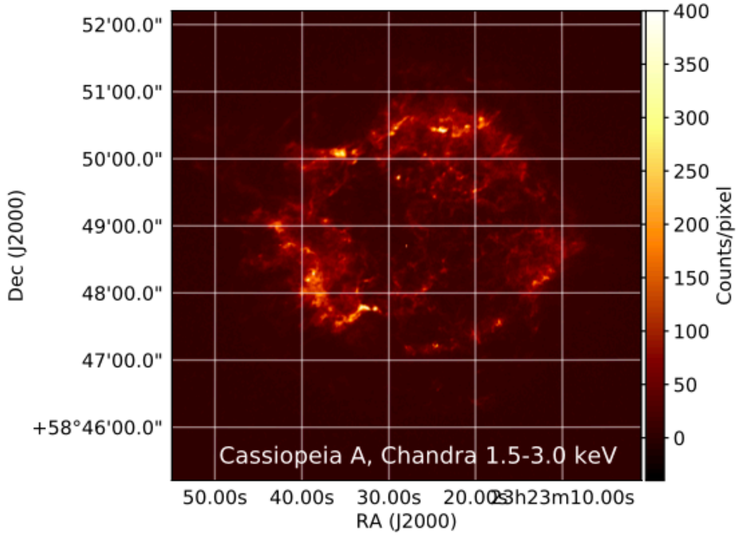
Architecture

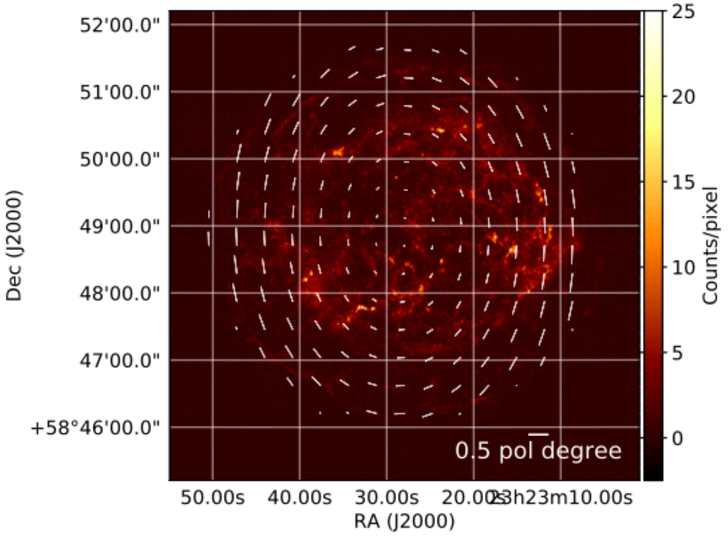


Simulation – Source Model definition

- Need to define essentially three properties:
  - Morphology (point sources, disks, annuli, generic extended sources from FITS images)
  - Energy spectrum in units of $[\mathrm{cm}^{-1}\,\mathrm{s}^{-1}\,\mathrm{keV}^{-1}]$
  - 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 in the same model
- Support for time-dependent transient and periodic sources







Simulation – Configuration file example

Example of a point source with background

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

# Pointing coordinates
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
PD = 0.1              # 10%
PA = 30.              # 30 degrees
SPEC = power_law(PL_NORM, PL_INDEX)
POL_DEG = constant(PD)
POL_ANG = constant(numpy.radians(PA))

# 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)
```

```
from ixpeobssim.srcmodel.spectrum import power_law
from ixpeobssim.srcmodel.polarization import constant
```

```
from ixpeobssim.srcmodel.roi import xPointSource, xROIModel
from ixpeobssim.srcmodel.bkg import xTemplateInstrumentalBkg
```

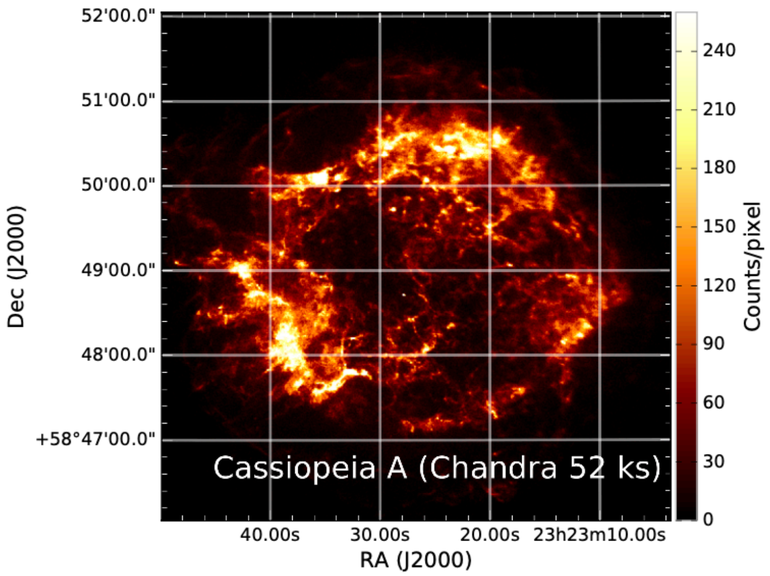
Simulation – Run script

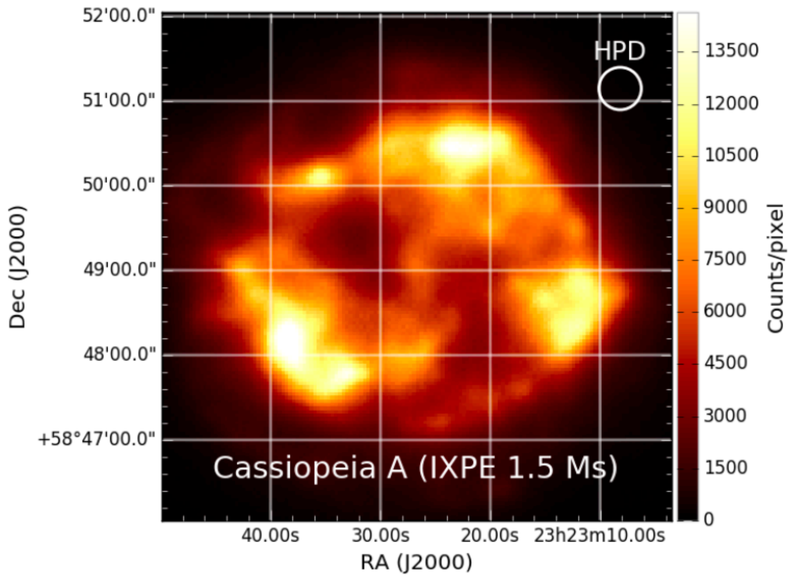
Example of a point source with background

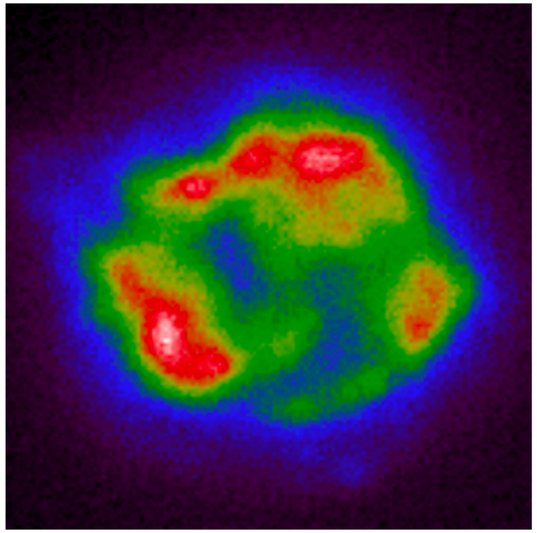
```
def simulate(duration=100000):
    """Run the simulation.
    """
    pipeline.xpobssim(duration=duration, configfile='toy_point_source_bkg.py')
```

- Simulate an observation starting from an arbitrary 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 more components the simulation is separately performed for each one and the resulting photon lists are then merged together.

Simulation – run the simulation







The simulation pipeline will generate 3 fits files, one for each DU.
Each file contains the simulated photon list and is analogous to the real-data files.

Analysis – select and bin

```
6  def select(src_rad=0.75, bkg_inner_rad=1.5, bkg_outer_rad=3.):
7      """Select the photon lists.
8      """
9      file_list = pipeline.file_list()
10     pipeline.xpselect(*file_list, rad=src_rad, suffix='src')
11     pipeline.xpselect(*file_list, innerrad=bkg_inner_rad, rad=bkg_outer_rad, suffix='bkg')
12
13 def bin_(ebinning=[2, 4, 8]):
14     """Create the necessary binned files.
15     """
16     pipeline.xpbin(*pipeline.file_list(), algorithm='CMAP')
17     kwargs = dict(algorithm='PCUBE', ebinalg='LIST', ebinning=ebinning)
18     pipeline.xpbin(*pipeline.file_list('src'), **kwargs)
19     pipeline.xpbin(*pipeline.file_list('bkg'), **kwargs)
20     for algorithm in ['PHA1', 'PHA1Q', 'PHA1U']:
21         pipeline.xpbin(*pipeline.file_list('src'), algorithm=algorithm)
22         pipeline.xpbin(*pipeline.file_list('bkg'), algorithm=algorithm)
```

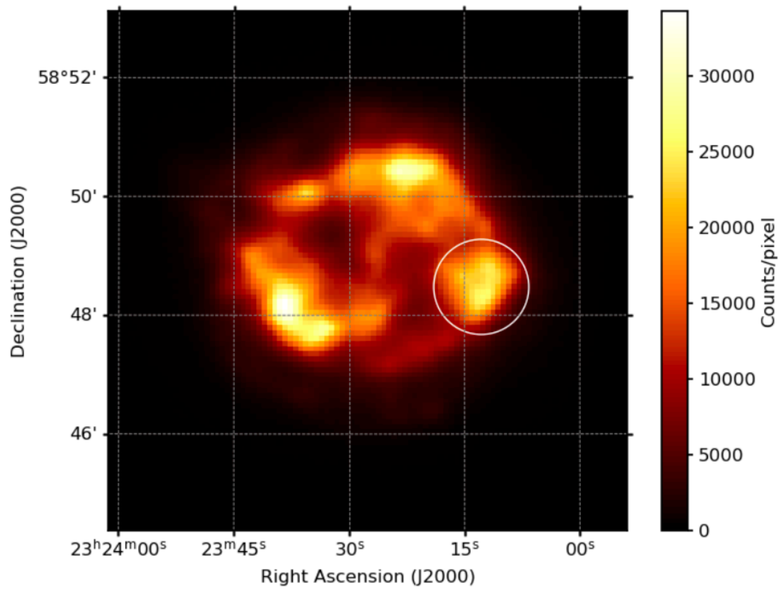
Analysis – select and bin

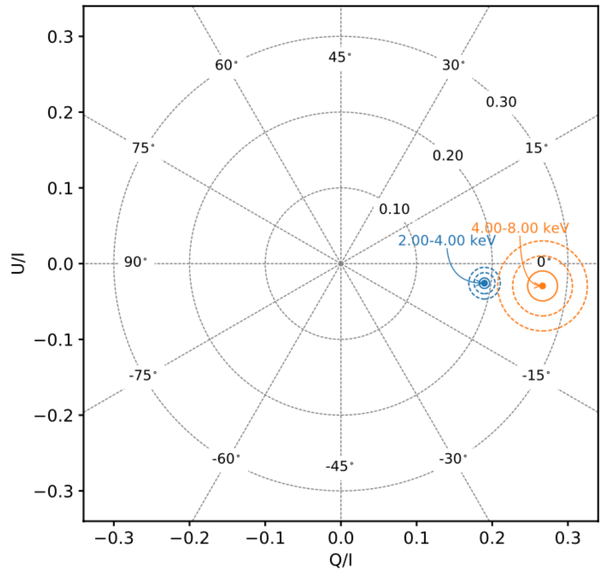
```
6  def select(src_rad=0.75, bkg_inner_rad=1.5, bkg_outer_rad=3.):
7      """Select the photon lists.
8      """
9      file_list = pipeline.file_list()
10     pipeline.xpselect(*file_list, rad=src_rad, suffix='src')
11     pipeline.xpselect(*file_list, innerrad=bkg_inner_rad, rad=bkg_outer_rad, suffix='bkg')
12
13 def bin_(ebinning=[2, 4, 8]):
14     """Create the necessary binned files.
15     """
16     pipeline.xpbin(*pipeline.file_list(), algorithm='CMAP')
17     kwargs = dict(algorithm='PCUBE', ebinalg='LIST', ebinning=ebinning)
18     pipeline.xpbin(*pipeline.file_list('src'), **kwargs)
19     pipeline.xpbin(*pipeline.file_list('bkg'), **kwargs)
20     for algorithm in ['PHA1', 'PHA1Q', 'PHA1U']:
21         pipeline.xpbin(*pipeline.file_list('src'), algorithm=algorithm)
22         pipeline.xpbin(*pipeline.file_list('bkg'), algorithm=algorithm)
```

Analysis – xpbin PCUBE

Polarization cube

- 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



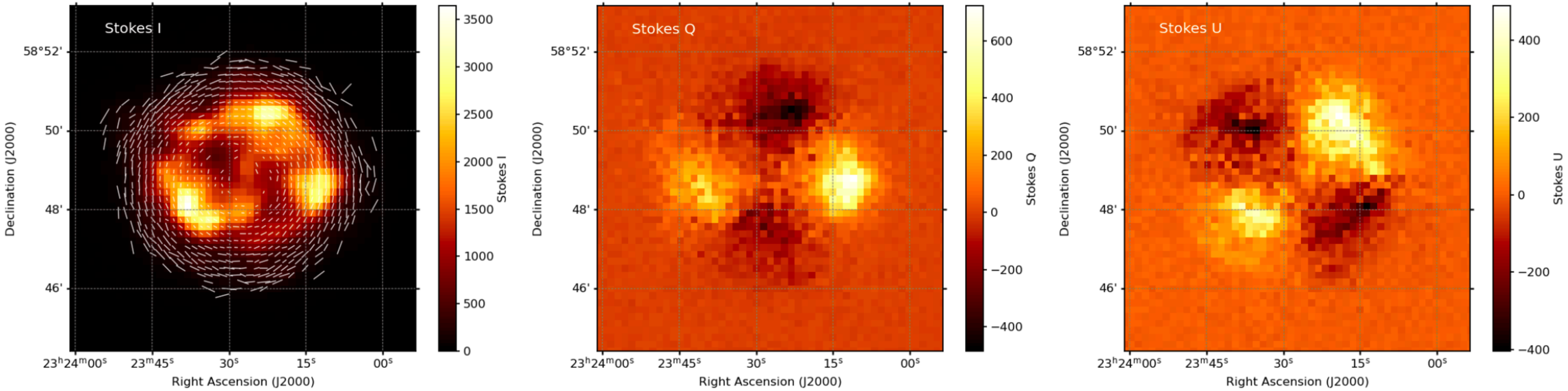


Quantity	2.00--8.00 keV
E_MEAN	5.0117599361664364
COUNTS	5762.0
MU	0.3921964538001015
W2	170.2692108154297
N_EFF	1664.503662109375
FRAC_W	0.28887602605160967
MDP_99	0.2681089106047689
I	532.3661499023438
I_ERR	13.048724174499512
Q	-67.53325653076172
Q_ERR	47.02301654580805
U	-61.832847595214844
U_ERR	47.02772717795586
QN	-0.12685489654541016
QN_ERR	0.08832833671794095
UN	-0.11614722013473511
UN_ERR	0.08833718519966481
QUN_COV	-8.85179443077969e-06
PD	0.17199519276618958
PD_ERR	0.0883654765108721
PA	-68.76152928032351
PA_ERR	14.73511598093247
P_VALUE	0.15021762132159794
CONFID	0.849782378678402
SIGNIF	1.035500479458798

Analysis – xpbin PMAP

Polarization map

- 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 the polarization information

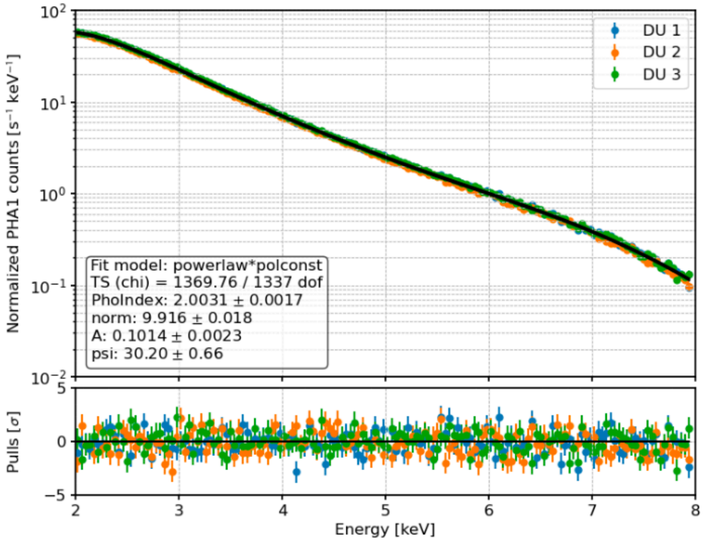


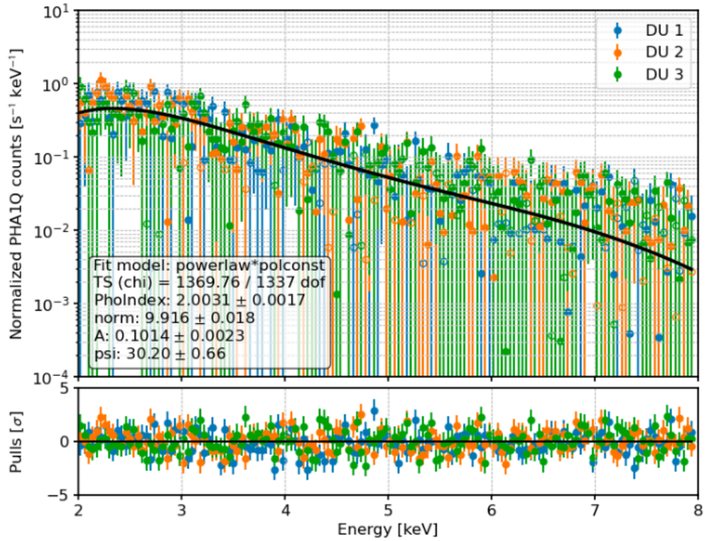
with the command `xpbinview` it is possible to quickly visualize all the outputs from xpbin

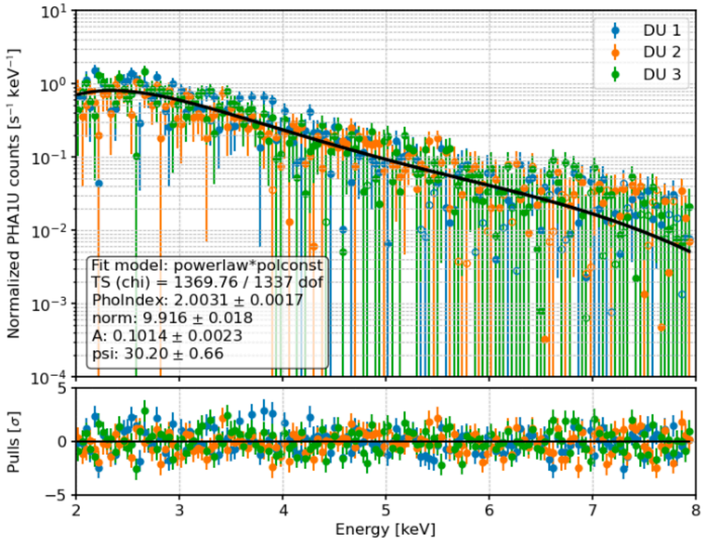
Analysis – xpbin PHA1, PHA1Q, PHA1U

Polarization map

- 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







References

Installation, overview, documentation, tutorials

Github repository

Please, cite the Paper
L. Baldini et al. 2022

Visit the ixpeobssim GitHub repository to contribute and submit issues you may find while using the software.

Take a look to the ixpeobssim tutorial material that was presented at HEAD20 meeting (March 2023)

In the near future

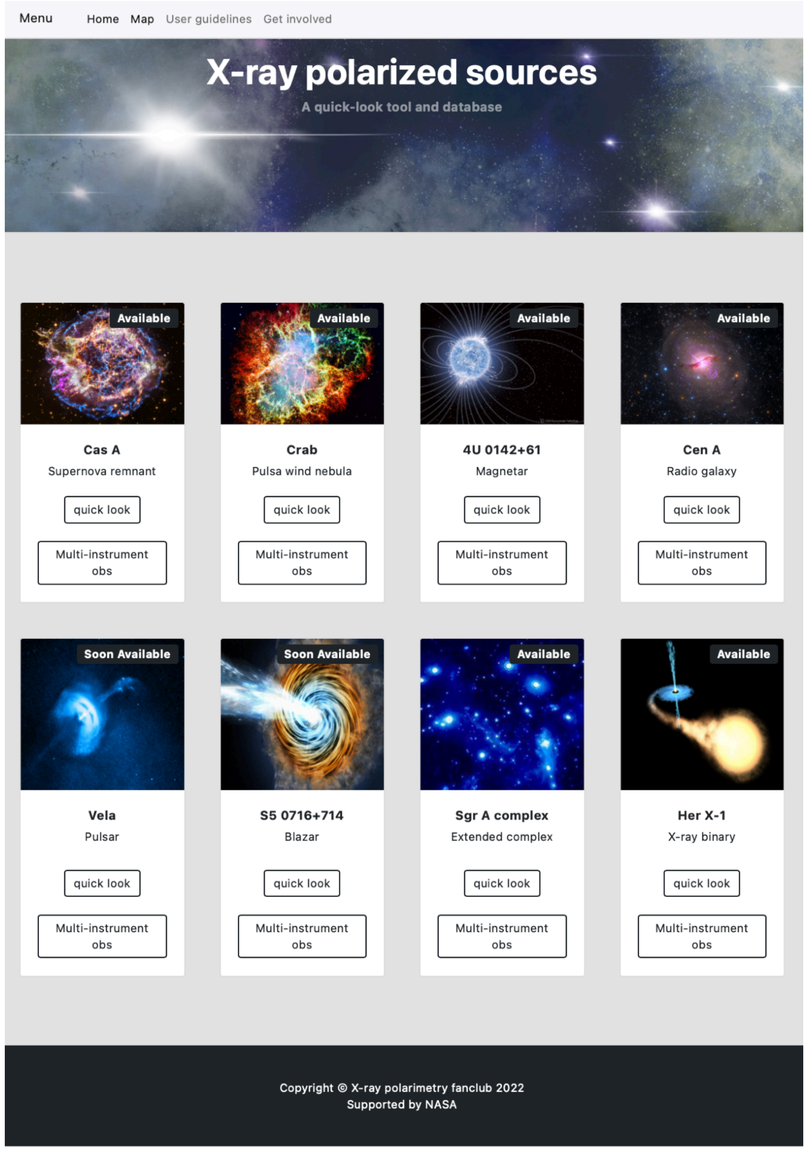
The IXPE quick-look tool and database

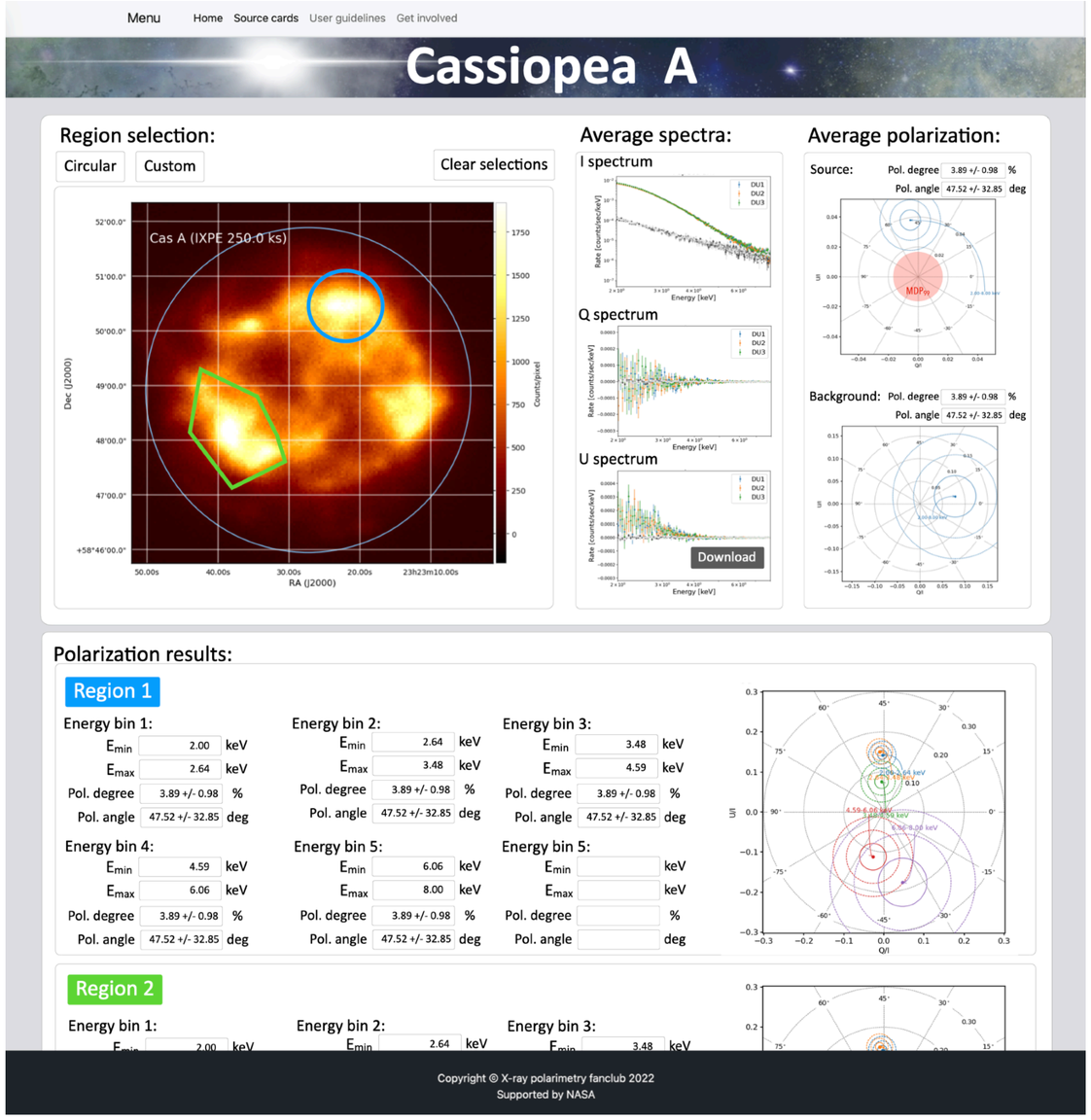
Stay tuned!

NASA-funded project through ADAP:

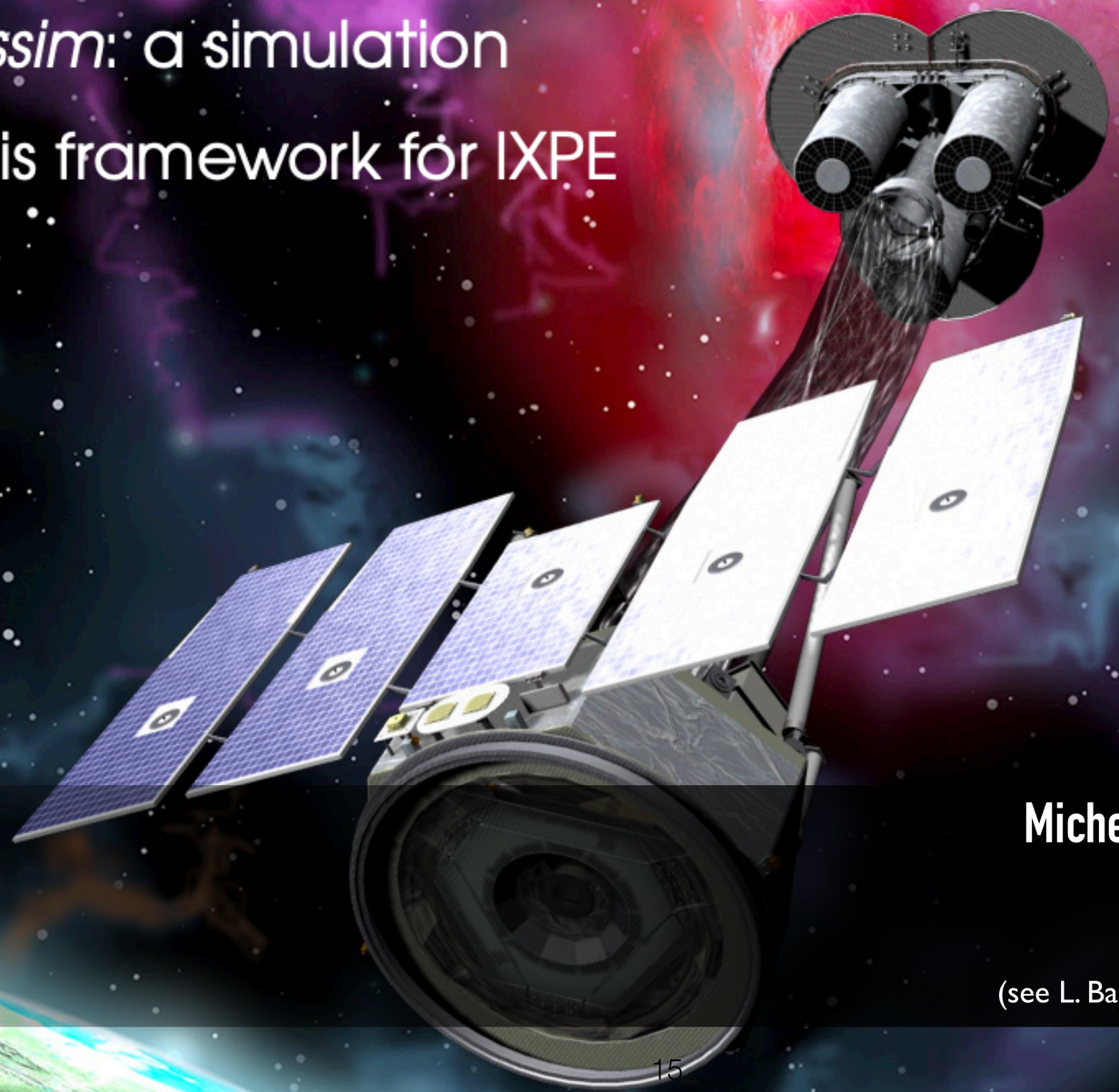
LSU
Stanford
NASA-MSFC

+ Collaborators from INFN-Pisa, INFN-Torino, LANL, University of Manitoba)





ixpeobssim: a simulation and analysis framework for IXPE



Michela Negro — Louisiana State University
Niccolò Di Lalla — Stanford University

Based on work done by a larger group
(see L. Baldidni et al., 2022, SoftwareX, Volume 19, id. 101194)

The project

- 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 upcoming General Observer program

Simulation – Several example provided

Installation, overview, documentation and tutorials are available:

If you download the package, you'll find examples of simulations at

`ixpeobssim/ixpeobssim/examples/`

Instrument Response Functions

- FITS files in OGIP format
- Each of the three IXPE DUs has its own set of IRFs
- Generated and stored in a local CALDB, then submitted to HEASARC