National Aeronautics and Space Administration
NASA
Fermi
Gamma-ray Space Telescope
www.nasa.gov/fermi

fermiPy: A New ST Analysis Scripting Tool

PyGamma15
November 18, 2015

Background and Motivation

- One of the goals of the Pass 8 effort was to provide tools to facilitate analyses that make use of the new Pass 8 event types

- **fermiPy** is a python software package that automates ST analysis with an emphasis on supporting Joint Likelihood analysis with two or more data selections

- Basic structure and design heavily borrowed from existing LAT analysis scripting tools (rungt, dsph pipeline, enrico, LATAnalysisScripts, etc.)
  - Analysis setup is defined with a text-based configuration file
  - Analysis execution is controlled with a class instance that executes the gt-tools and uses the pyLikelihood ST interface for model building and optimization

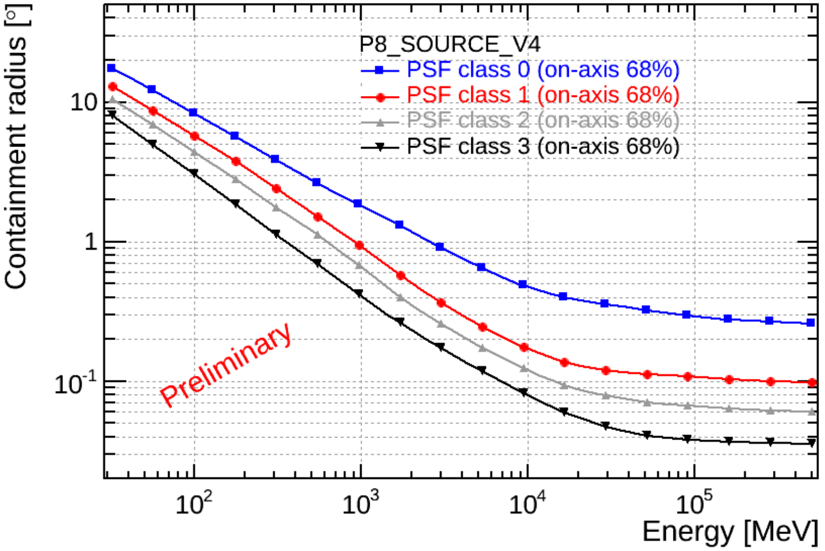
- More information:
  - **Documentation**: http://fermipy.readthedocs.org/en/latest/index.html
  - **GitHub Page**: https://github.com/fermiPy/fermipy

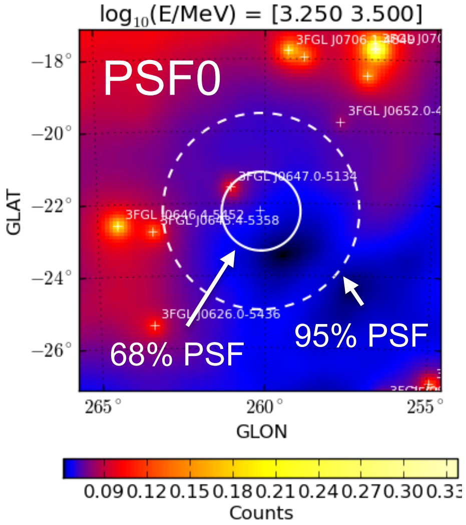
Joint Likelihood with PSF Event Types

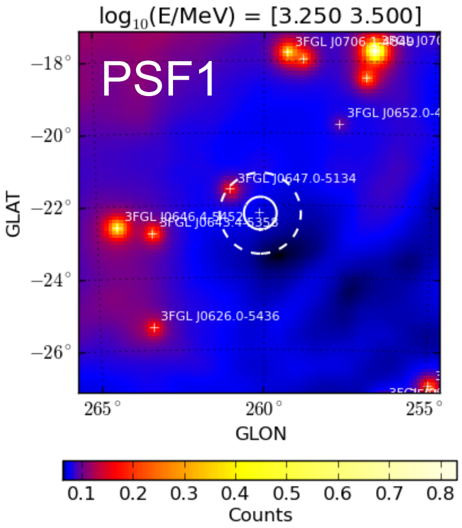
Joint likelihood w/ PSF event types weights events according to the quality of their angular reconstruction

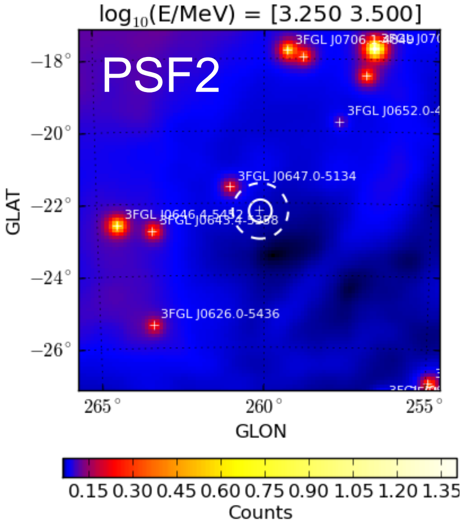
Joint analysis improves point-source sensitivity by at least ~10% with respect to a combined analysis (evtype=3)

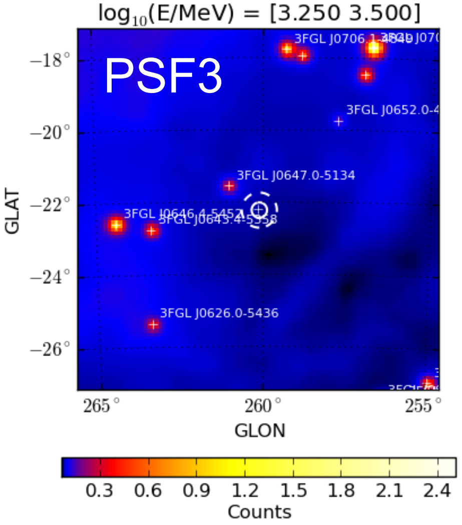
Larger gains expected for analyses in regions with significant source confusion and measurements of angular extension











Configuration File

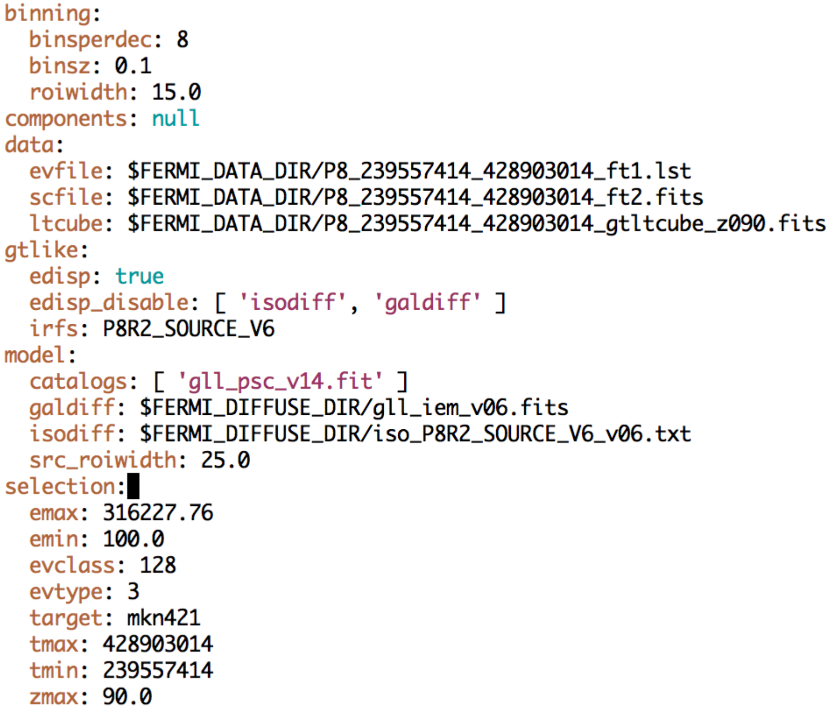
- Configuration is controlled with YAML files
  - Human-readable format with data structures that parallel those in python (scalars, lists, dictionaries)
  - Easy reading/writing of config files with PyYAML module

- Contents of the configuration file define a structured hierarchy of parameters that maps to a python configuration dictionary

- Each configuration block (binning, data, model, etc.) groups a set of related parameters

Sample Configuration

P8 SOURCE class with 15x15 deg ROI
and 8 bins per decade



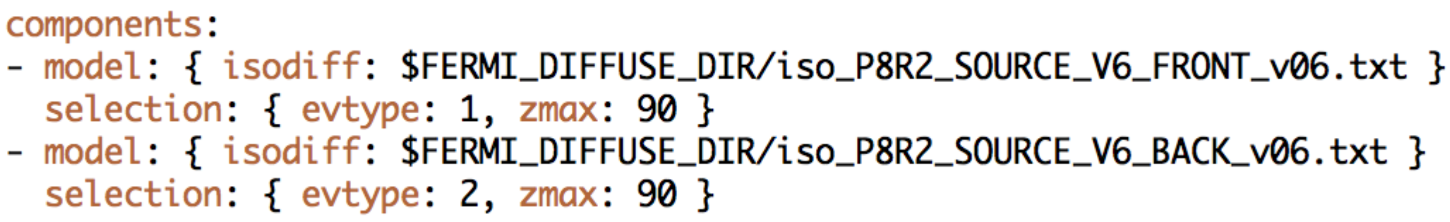
```
binning:
  binsperdec: 8
  binsz: 0.1
  roiwidth: 15.0
components: null
data:
  evfile: $FERMI_DATA_DIR/P8_239557414_428903014_ft1.lst
  scfile: $FERMI_DATA_DIR/P8_239557414_428903014_ft2.fits
  ltcube: $FERMI_DATA_DIR/P8_239557414_428903014_gtltcube_z090.fits
gtlike:
  edisp: true
  edisp_disable: [ 'isodiff', 'galdiff' ]
  irfs: P8R2_SOURCE_V6
model:
  catalogs: [ 'gll_psc_v14.fit' ]
  galdiff: $FERMI_DIFFUSE_DIR/gll_iem_v06.fits
  isodiff: $FERMI_DIFFUSE_DIR/iso_P8R2_SOURCE_V6_v06.txt
  src_roiwidth: 25.0
selection:
  emax: 316227.76
  emin: 100.0
  evclass: 128
  evtype: 3
  target: mkn421
  tmax: 428903014
  tmin: 239557414
  zmax: 90.0
```

Components Block

- By default fermiPy will use a single data selection and likelihood

- The **components** block can be used to define a joint likelihood using two or more independent sub-selections of the data (e.g. Front/Back, PSF types)
  - Components block can be organized as either a list or dictionary of analysis configurations
  - Configuration for each component shares the same structure as the root analysis (i.e. with sub-blocks for data, model, selection, etc.)
  - Any parameter not defined for a given component defaults to the value defined in the root analysis

Sample Components Block for Front/Back Analysis

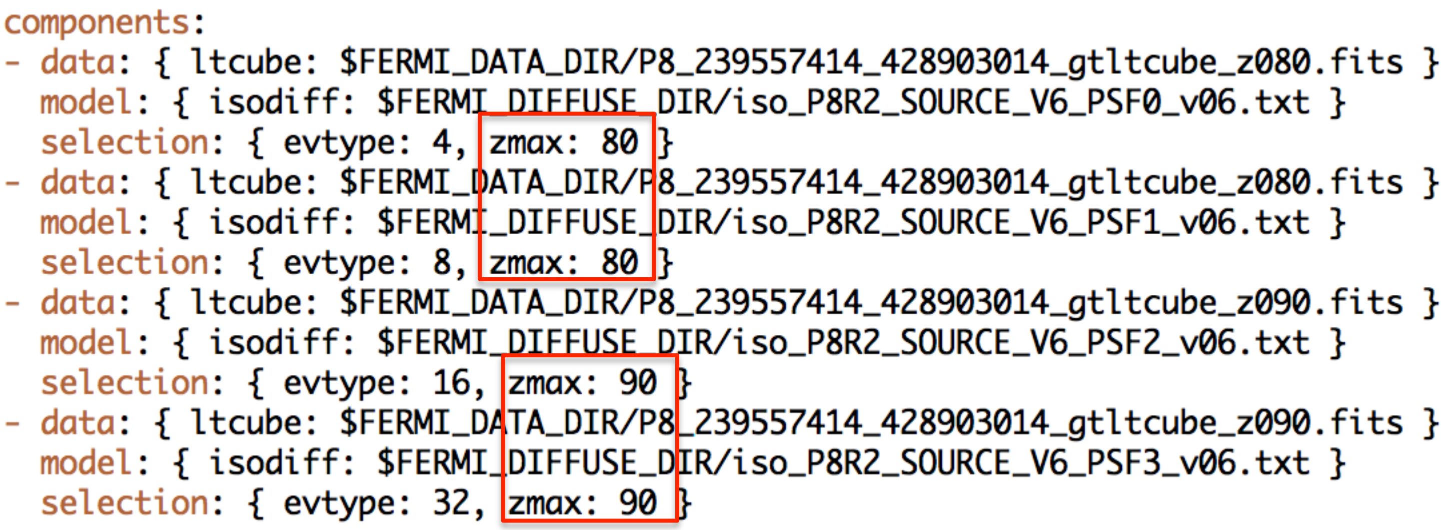


```
components:
- model: { isodiff: $FERMI_DIFFUSE_DIR/iso_P8R2_SOURCE_V6_FRONT_v06.txt }
  selection: { evtype: 1, zmax: 90 }
- model: { isodiff: $FERMI_DIFFUSE_DIR/iso_P8R2_SOURCE_V6_BACK_v06.txt }
  selection: { evtype: 2, zmax: 90 }
```

A More Complex Example

- Components can be fully customized with respect to any of the parameters in the root configuration
  - Zenith selections
  - Energy ranges
  - Spatial/Energy Bin Size
  - Models

Sample Components Block for Analysis with PSF0-PSF3 Types



```
components:
- data: { ltcube: $FERMI_DATA_DIR/P8_239557414_428903014_gtltcube_z080.fits }
  model: { isodiff: $FERMI_DIFFUSE_DIR/iso_P8R2_SOURCE_V6_PSF0_v06.txt }
  selection: { evtype: 4, zmax: 80 }
- data: { ltcube: $FERMI_DATA_DIR/P8_239557414_428903014_gtltcube_z080.fits }
  model: { isodiff: $FERMI_DIFFUSE_DIR/iso_P8R2_SOURCE_V6_PSF1_v06.txt }
  selection: { evtype: 8, zmax: 80 }
- data: { ltcube: $FERMI_DATA_DIR/P8_239557414_428903014_gtltcube_z090.fits }
  model: { isodiff: $FERMI_DIFFUSE_DIR/iso_P8R2_SOURCE_V6_PSF2_v06.txt }
  selection: { evtype: 16, zmax: 90 }
- data: { ltcube: $FERMI_DATA_DIR/P8_239557414_428903014_gtltcube_z090.fits }
  model: { isodiff: $FERMI_DIFFUSE_DIR/iso_P8R2_SOURCE_V6_PSF3_v06.txt }
  selection: { evtype: 32, zmax: 90 }
```

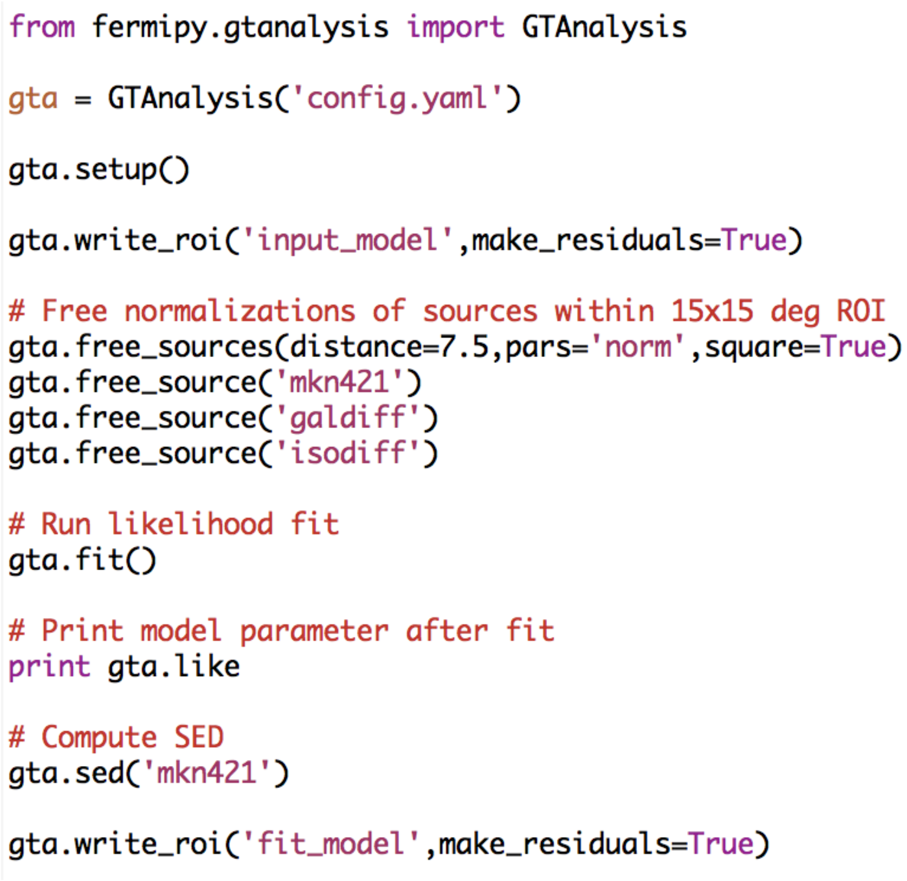
Using fermiPy

- fermiPy is primarily designed as a set of modules that can be integrated into existing analysis scripts

- Analysis is executed through the methods of the GTAnalysis class

- Underlying pyLikelihood classes are exposed through the **like** property which points to an instance of SummedLikelihood

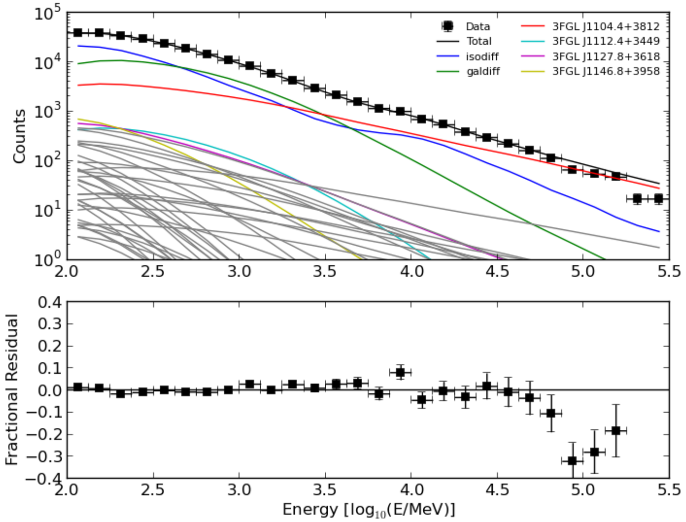
Sample Analysis Script



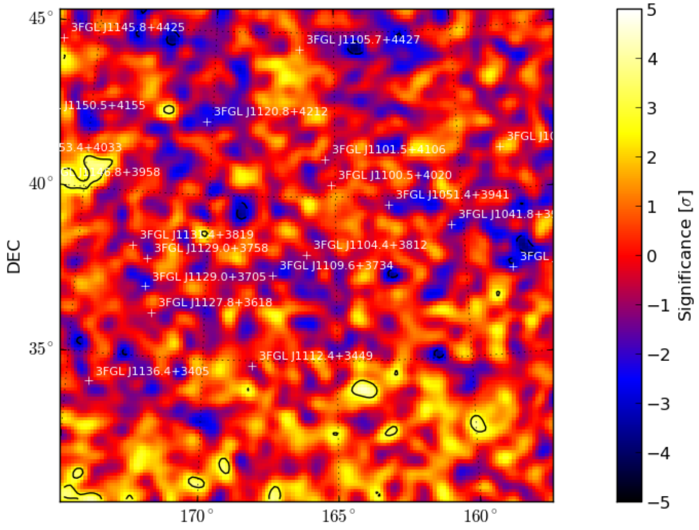
```
from fermipy.gtanalysis import GTAnalysis

gta = GTAnalysis('config.yaml')

gta.setup()

gta.write_roi('input_model',make_residuals=True)

# Free normalizations of sources within 15x15 deg ROI
gta.free_sources(distance=7.5,pars='norm',square=True)
gta.free_source('mkn421')
gta.free_source('galdiff')
gta.free_source('isodiff')

# Run likelihood fit
gta.fit()

# Print model parameter after fit
print gta.like

# Compute SED
gta.sed('mkn421')

gta.write_roi('fit_model',make_residuals=True)
```

Sample Analysis: Mkn 421

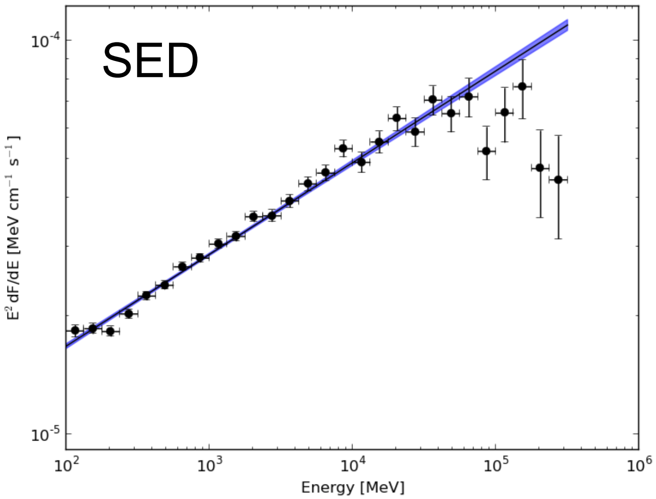
Counts Spectrum



Spatial Residuals

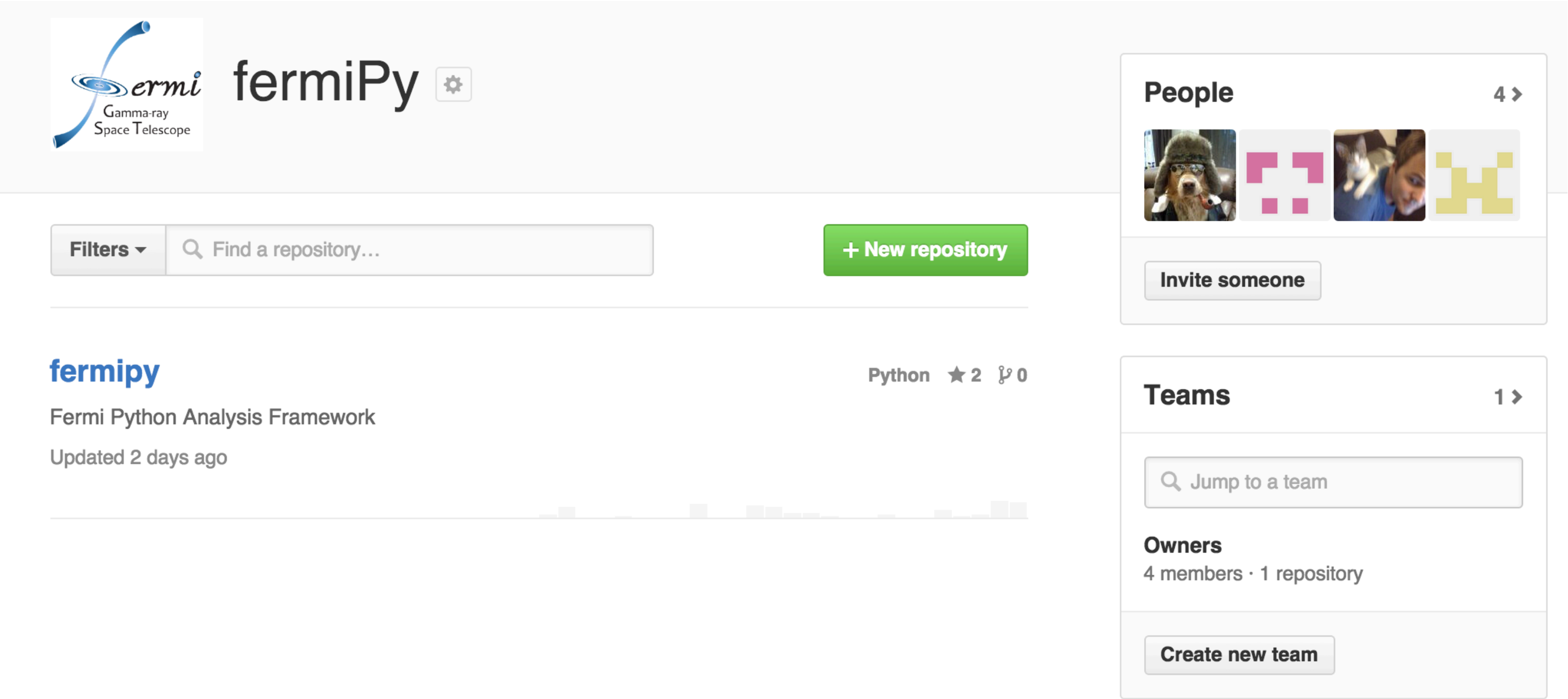


P8 Source Class
15x15 deg ROI
6 years
4 PSF Event Types
PSF0/1: zmax=80
PSF2/3: zmax=90



How to Contribute

- Contributions are welcome!
- If you are interested in contributing code please ask to be invited to the fermiPy developer list
- Feedback and feature requests are also welcome



fermiPy

fermipy

Fermi Python Analysis Framework

Updated 2 days ago

Python ★ 2 0

People 4

Teams 1

Owners

4 members · 1 repository