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

fermitools and *fermipy*

Eric Charles
SLAC

PyGamma 2019 Workshop
Heidelberg, Germany
18-22 March 2019

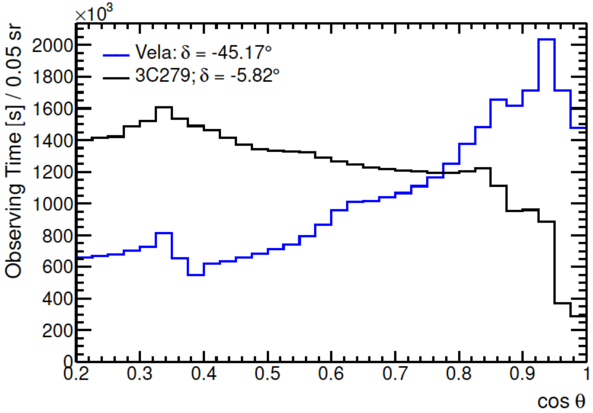
Outline

- Focus on a few aspects of the *fermitools* and *fermipy* environment
  - *fermitools*
  - *fermipy* Region of Interest (ROI)-based analysis
    - Examples of functionality
  - *fermipy* analysis pipelines
    - Stacked Dark Matter search with *dmpipe* analysis pipeline
- Summary

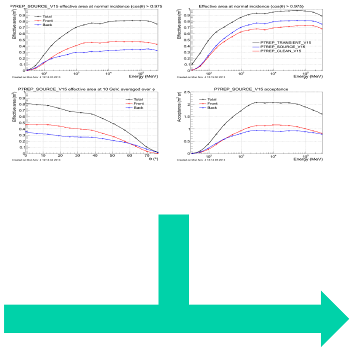
Fermitools Standalone Applications

See talks by Perkins, Asercion

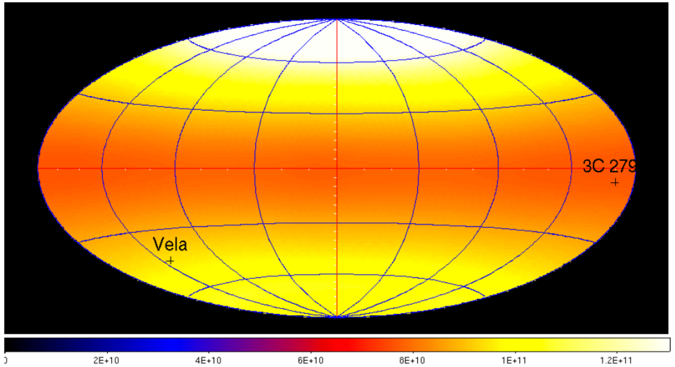
Observing Profile, t(θ; α, δ) produced by *gtltcube*



IRFs: A_{eff}, PSF, E_{disp}



Exposure Map, E(E; α, δ) produced by *gtexpcube2*

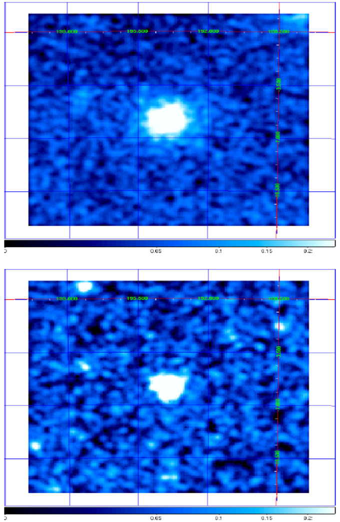


- *fermitools* include a set of standalone applications to perform each stage of a standard likelihood analysis
- *fermitools* supported by the Fermi Science Support Center (FSSC)

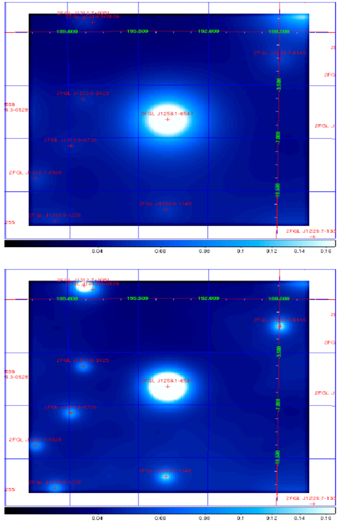
Likelihood Analysis with *fermitools*

See talk by Perkins

Data maps, N(E; α, δ) produced by *gtbin*



Model maps, M(E; α, δ) produced by *gtmodel*



Binned likelihood analysis (e.g., *gtlike*) optimizes model by maximizing the Poisson likelihood:

$$-\ln\mathcal{L} = \sum_{i}^{\text{pix}} \sum_{k}^{\text{energy}} N_{ik} \ln M_{ik} - M_{ik},$$

fermitools python Interface

- The standalone *fermitools* implement a simple likelihood analysis:
  - Binned or unbinned data
  - Fits a single data “component”,
    - Single region
    - Single event class
  - Fitting is based on spatial templates, re-localizing sources or changing morphologies is not generally supported
  - Optimize the model provided, user intervention is required for further optimization
- Python interface into the underlying libraries allow user to avoid these limitations

fermipy Analysis Package

- Written by Matthew Wood inspired by the previous pygamma meeting
- GitHub package
  - https://github.com/fermiPy/fermipy
- Available in pip and conda (latest release: fermipy-0.17.4)
- Increasingly becoming standard for high-level Fermi-LAT analysis
  - 60k+ conda downloads (including download for automatic testing)
  - Used in Fermi summer school run by the FSSC
- Several example jupyter notebooks available
  - https://github.com/fermiPy/fermipy-extra/blob/master/notebooks

Fermipy Implements Region-Based Analysis

- Key concept is that *fermipy* analysis is "region" based:
  - *fermipy* provides a set of tools to model the gamma-ray emission in a region of interest (ROI)
- Produces standard analysis products
- Includes "context", such as adding sources from a catalog
- Includes hooks to manipulate sources:
  - add & remove sources from the model
  - change model and model parameters of any source
  - fix and free model parameters
- Includes "meta" tools, such as optimizing a region by iteratively fitting the region with different sources fixed and freed

fermipy Analysis Example

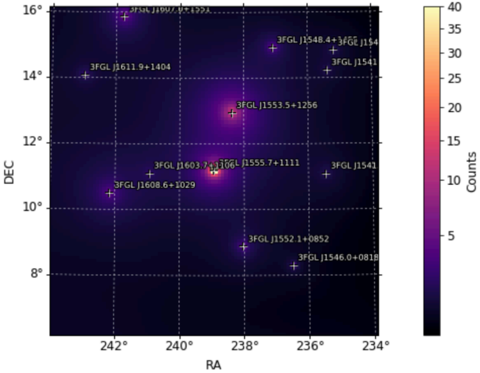
```
from fermipy.gtanalysis import GTAnalysis
gta = GTAnalysis(‘config.yaml’)
gta.setup()
gta.write_roi(‘initial’)
gta.optimize()
gta.write_roi(‘baseline’, make_plots=True)
```

- Some notes:
  - ‘config.yaml’ is a 25 line (or more) yaml configuration file, specifying dataset, region of interest, binning parameters, and data analysis options
  - gta.write_roi() writes a snapshot of the analysis region, containing everything needed to reproduce exactly the model of the region

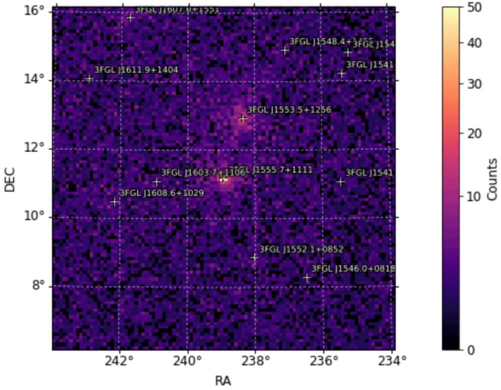
Baseline Analysis is Already a Major Task

- The six line script on the previous slide:
  - Constructs all the data products need to analysis the region of interest (ROI)
  - Builds a model of the ROI, including catalog sources and diffuse emission models
  - Iteratively optimizes the ROI model
  - Makes a set of diagnostic plots

Model map, n(E; α, δ)

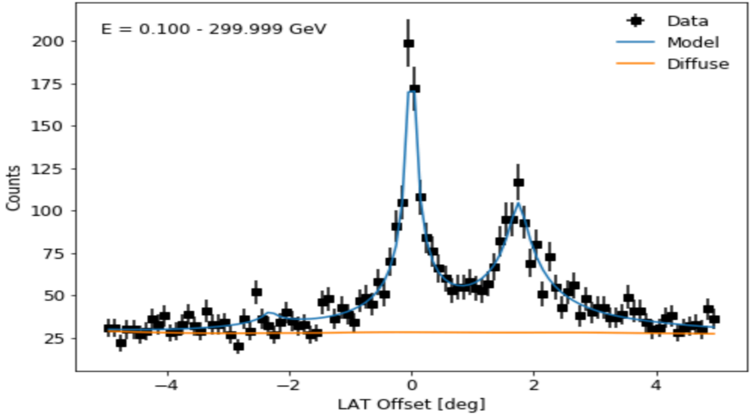


Data map, n(E; α, δ)

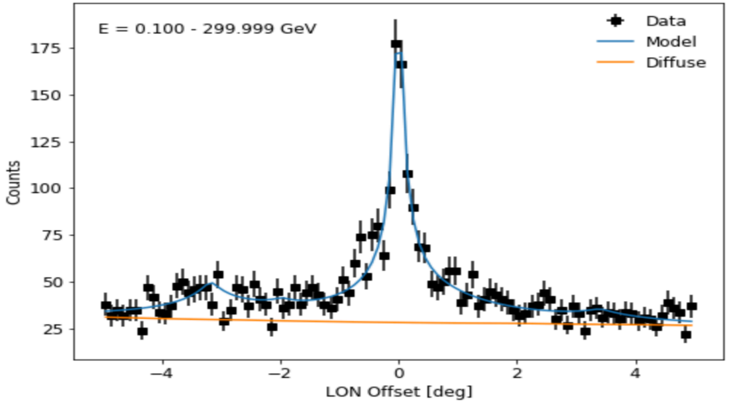


Diagnostics: Projected Data / Model Comparisons

Data & Model, Latitude Projection



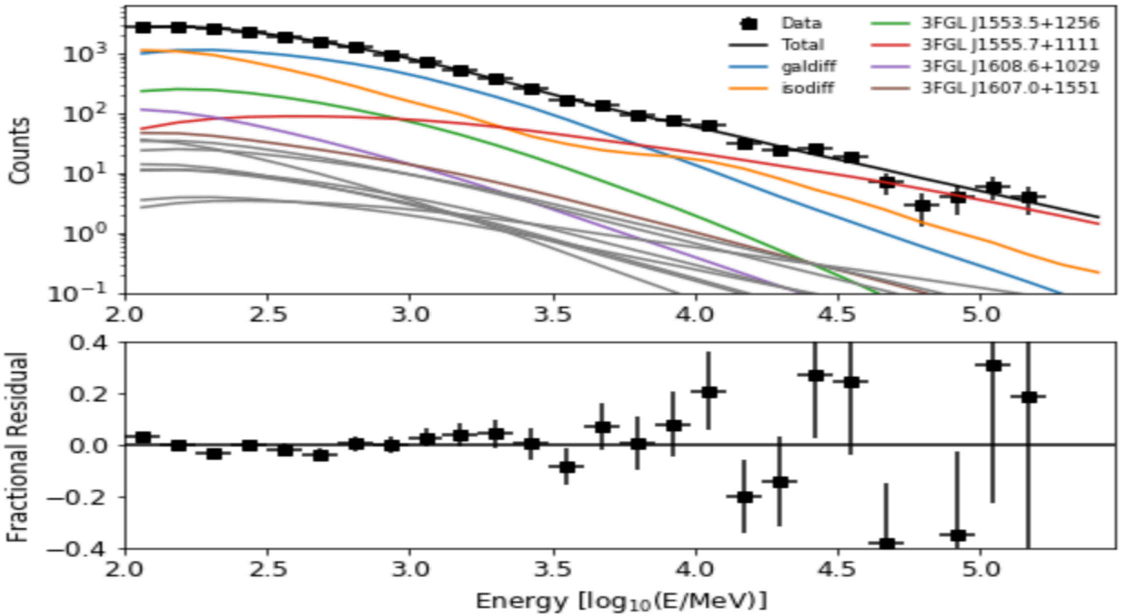
Data & Model, Longitude Projection



- Fermipy can also produce a number of standard diagnostic plots
  - E.g., X-Y projections of counts in region with data / model comparison
  - Easy to extend or customize, e.g., adding curves for particular sources, or restricting to specific energy bands

Diagnostics: Counts Spectra Modeling

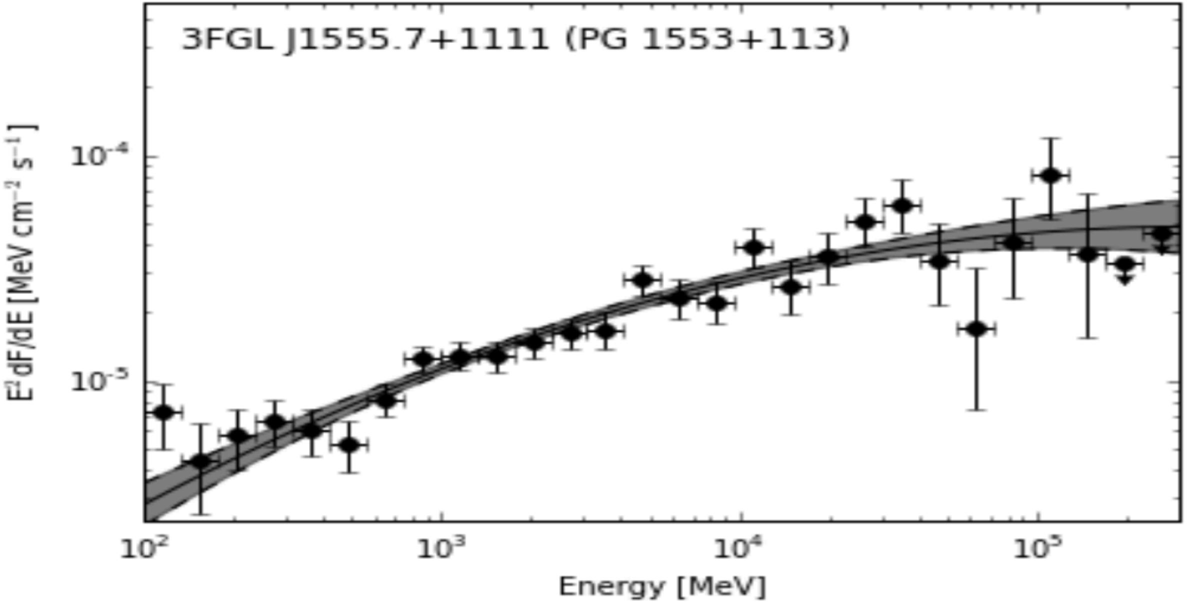
Observed Counts, separated by source



- Contributions to observed counts from all sources in the ROI model
- Plots are configurable, can select sub-regions of ROI

Spectral Energy Density

Spectral Fit, including bin-by-bin fluxes and broadband fit envelope

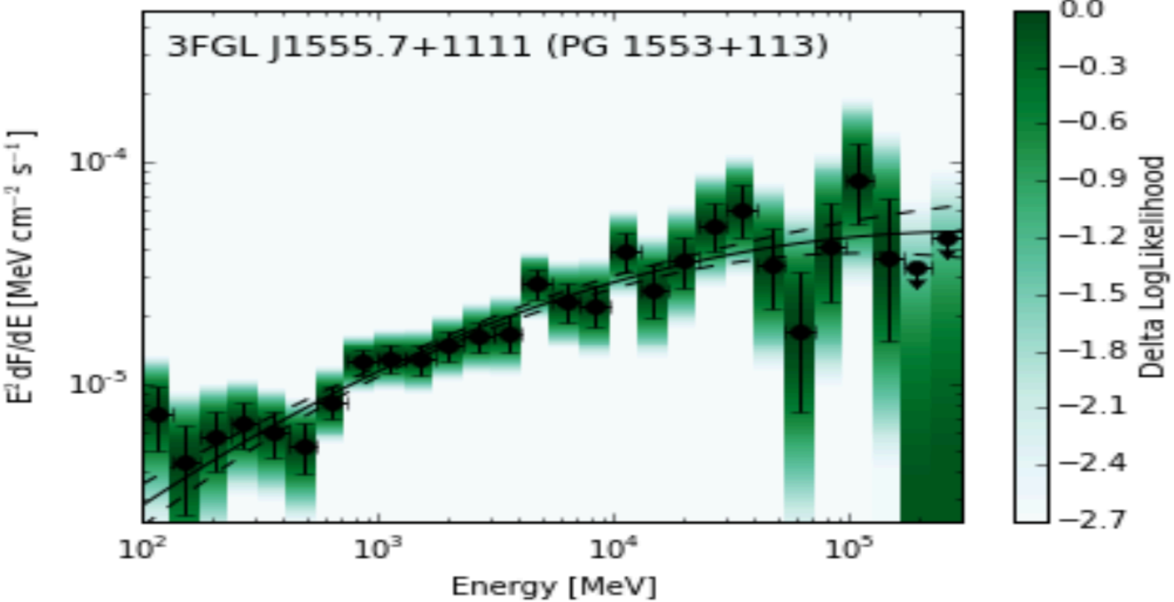


- Also, *fermipy* makes it very simple to fit the spectra of individual sources

```
gta.sed('3FGL J1555.7+1111', make_plots=True)
```

"Castro" Likelihood Curves

Spectral Fit, including bin-by-bin fluxes, likelihoods and broadband fit envelope



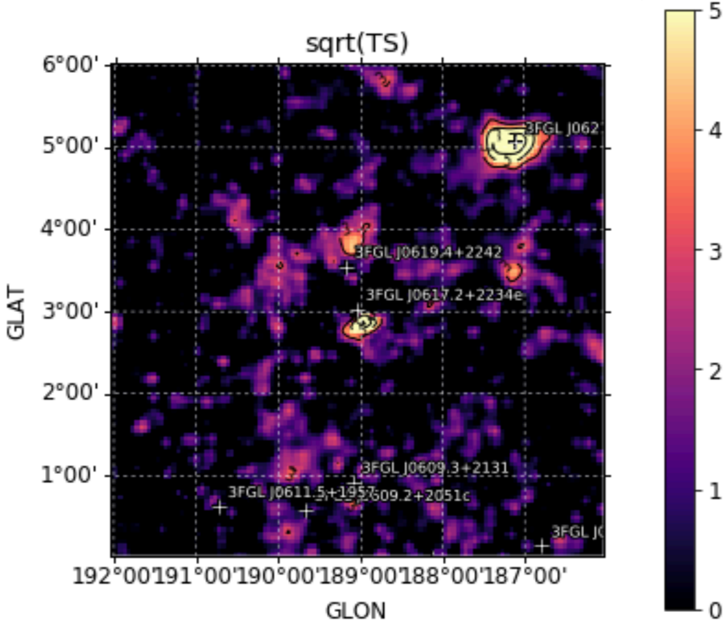
- Complete likelihood curve is much more useful that error bars / upper limits, so we developed a format to store them:
- https://gamma-astro-data-formats.readthedocs.io/en/latest/spectra/binned_likelihoods/index.html

Source Finding, Test Statistic Maps

Source finding Algorithm builds test statistic (TS) map for region and identifies peaks

$$TS = 2\frac{\mathscr{L}_S}{\mathscr{L}_0}$$

TS map w.r.t. baseline model of region



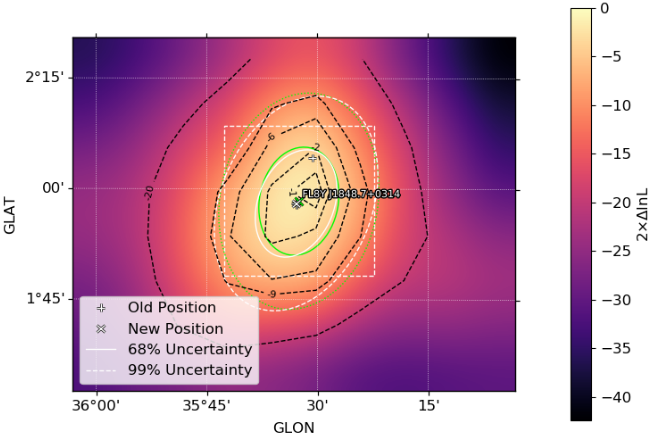
Note: one always builds a TS map with respect a baseline model of a region that is treated as the “null hypothesis”

- Also, very simple to find new sources in the region:

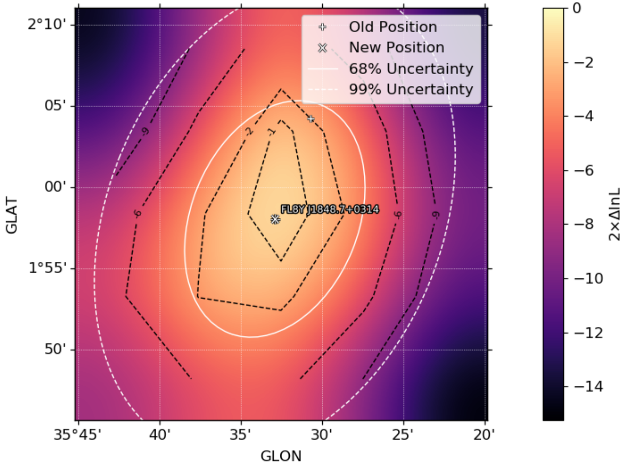
```
new_srcs = gta.find_sources(sqrt_ts_threshold=5,min_separation=0.2)
```

Localizing Sources

“Fast” 2° x 2° TS map



“Full” 5 x 5 pixel TS map



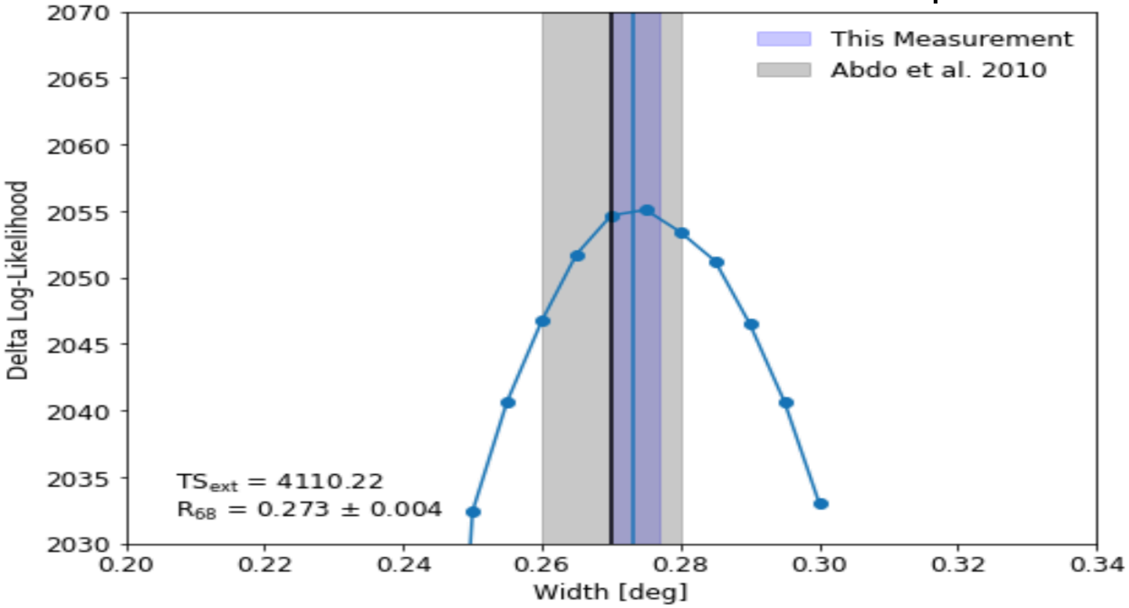
- Also, very simple to re-localize any source:

```
gta.lcoalize(srcName)
```

- Two-step process:
  - Wider “fast” TS-map with background fixed, approximate PSF image
  - Zoomed “full” TS-map with background free, exact PSF image

Source Extension

Profile likelihood scan of source extension parameter



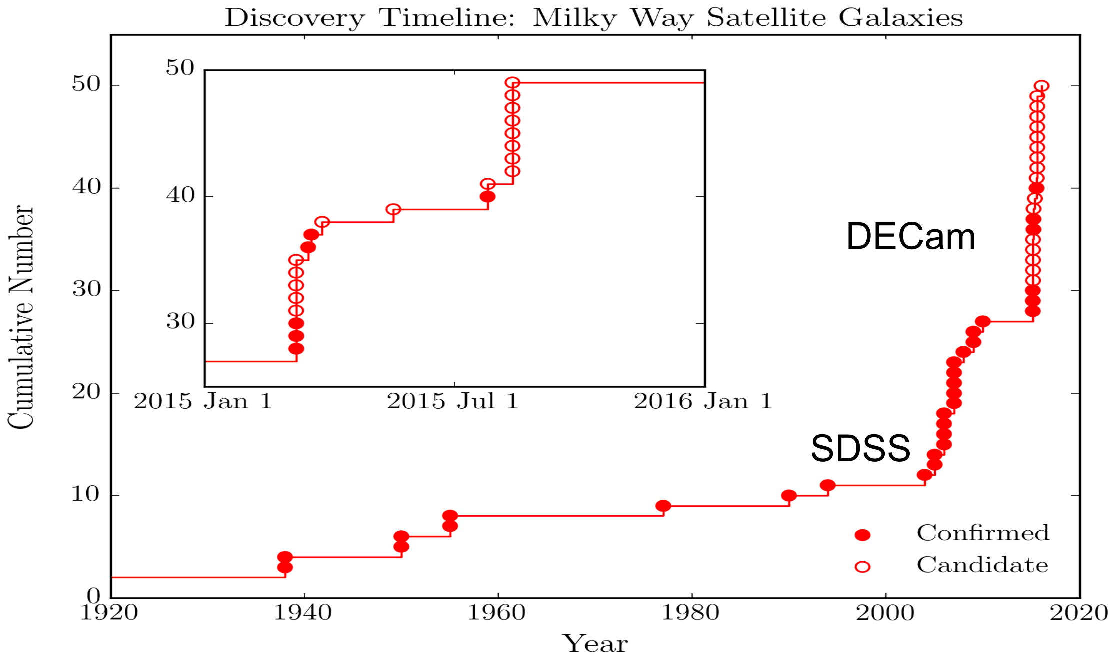
- *fermipy* can fit source extension by doing profile likelihood scan of extension parameter

```
gta.extension(srcName)
```

fermipy analysis pipelines

- Fermi-LAT experience has shown us that users end up producing fairly involved data analysis pipelines
  - Analysis complexity expands to fill available resources
- *fermipy* includes simple workflow tools implemented in *fermipy.jobs,* used to implement:
  - Stacked Dark Matter searches (*dmpipe*)
  - All-sky diffuse emission fitting (*fermipy.diffuse*)
- Example jupyter notebook using *dmpipe*:
  - https://github.com/fermiPy/fermipy-extra/blob/master/notebooks/dSphs.ipynb

Rapidly Growing Number of Targets (eg., dSphs)



DES Year 2 Data: Drlica-Wagner+(2015)

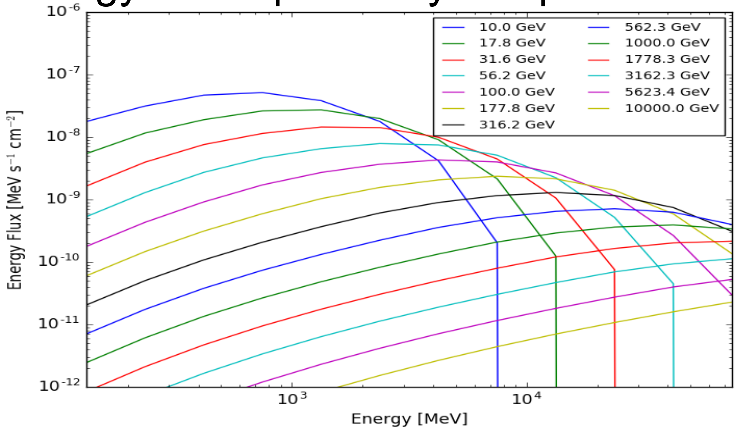
DES Year 1 Data: Bechtol+ (2015)

Koposov+ (2015)

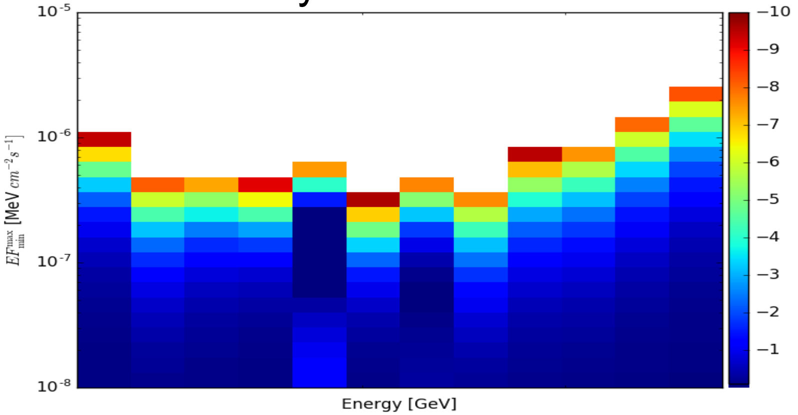
- Advent of deep, digital survey era in optical astronomy has led to the discovery of numerous new Milky Way-satellite dwarf galaxies
- LSST & other surveys will continue to find new dwarf galaxies after the *Fermi* mission

Stacking Analysis Requires a “Stacking” Variable

Energy Flux Spectra by DM particle mass



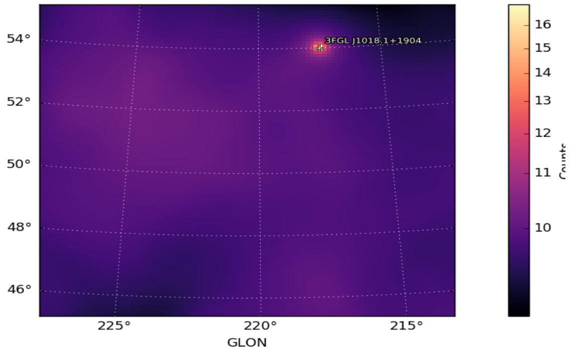
SED bin-by-bin likelihood values



- Stacking (i.e., analysis of multiple targets) analyses common in γ-ray astronomy
- Key to proper treatment of stacking, creating a model the exactly what the stacked targets have in common, e.g.,:
  - For DM searches, the interaction cross section <σv>
  - For radio galaxies, radio-gamma flux correlation coefficient

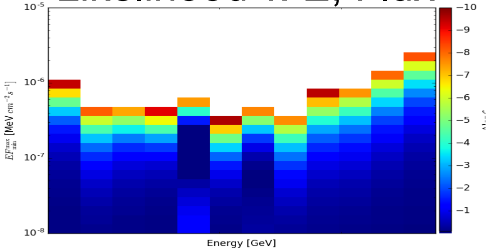
Example: DM Analysis Pipeline

Single Target
ROI Baseline Analysis



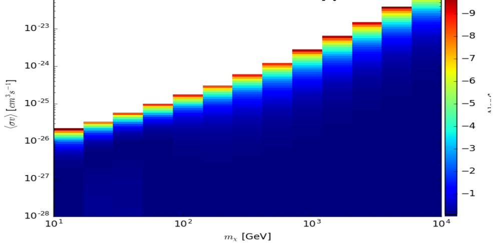
fermipy-analyze-sed

Target SED
Likelihood v. E, Flux

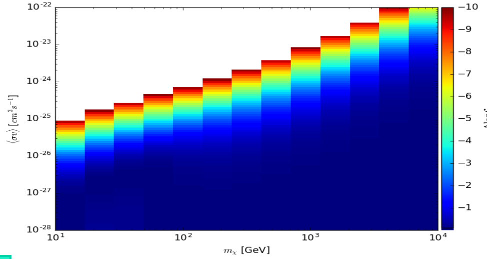


dmpipe-convert-castro

Target DM
Likelihood v. m_χ, <σv>

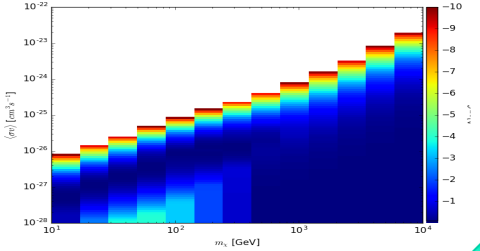


Target DM Likelihood
w. J-factor uncertainties



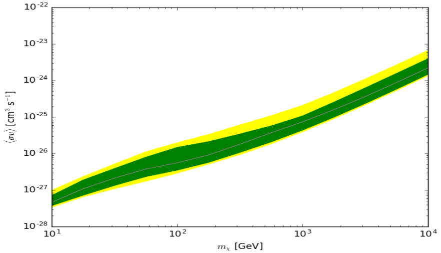
dmpipe-stack-likelihood

Stacked DM Likelihood
for all multiple dSphs



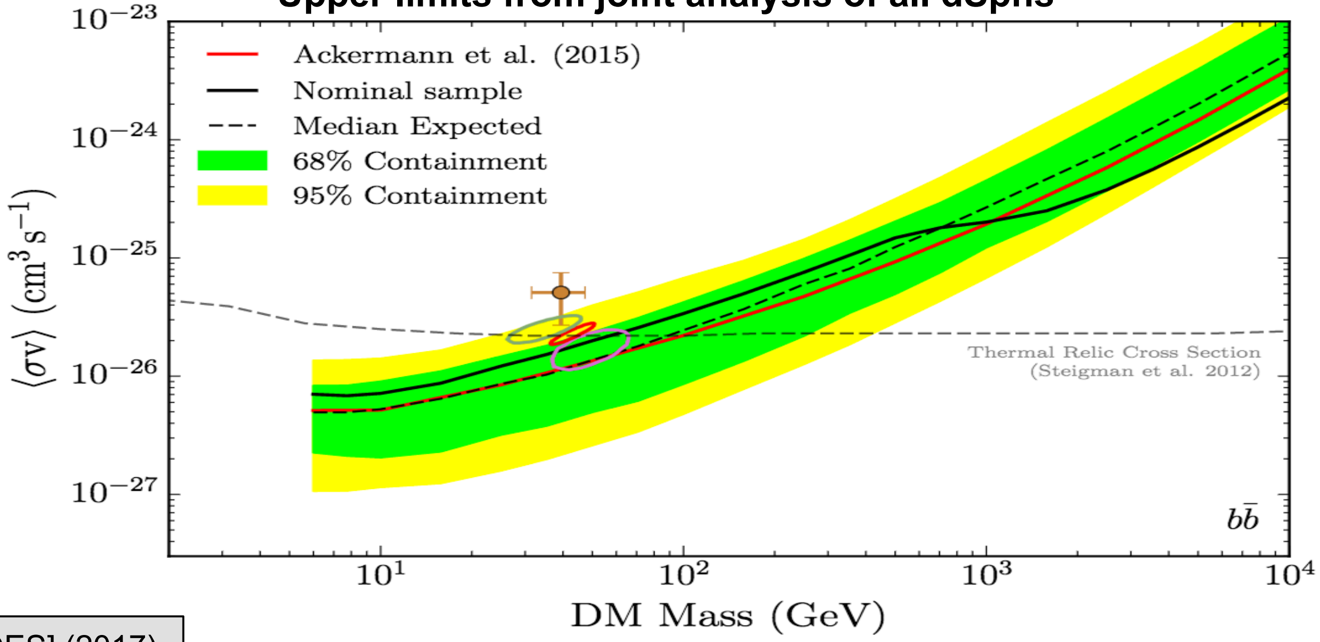
dmpipe-collect-stacked-limits

Expected upper limits
from null-control simulations



Joint Fitting Results from dSphs

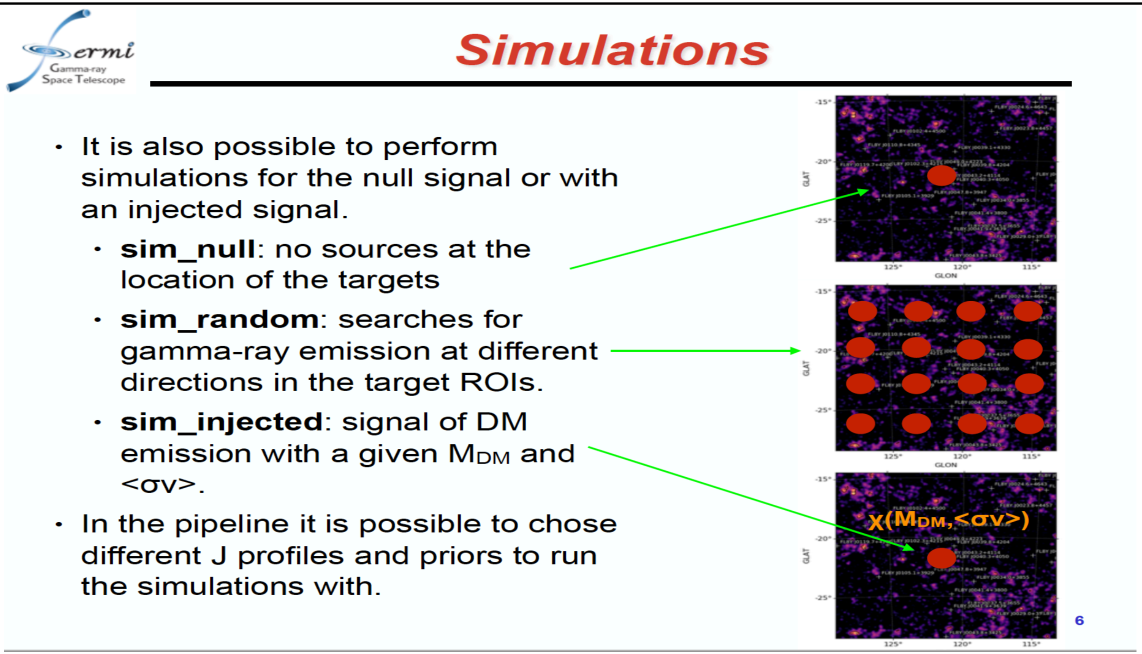
Upper limits from joint analysis of all dSphs



Albert+ [LAT & DES] (2017)

- Stacked analysis of a “nominal sample” of confirmed dSphs is well within the expectation band for null hypothesis and is in mild tension with DM interpretations of GC excess

Standard Control Simulations



Simulations

- It is also possible to perform simulations for the null signal or with an injected signal.
  - **sim_null**: no sources at the location of the targets
  - **sim_random**: searches for gamma-ray emission at different directions in the target ROIs.
  - **sim_injected**: signal of DM emission with a given M_{DM} and $<\sigma v>$.
- In the pipeline it is possible to chose different J profiles and priors to run the simulations with.

6

- A significant (dominant?) part of analysis is performing standard control studies
- We have implemented standardized version of these in *dmpipe*
  - Fast simulations, throw Poisson noise on model map of ROI

Summary

- *fermitools* provide libraries and standalone applications to perform Fermi-LAT data analysis
- Many users have scripted *fermitools* to implement high-level analysis
- *fermipy* provides a region-based analysis framework
  - Functionality to optimize model of region in a variety of ways
- *fermipy.jobs* provides pipeline-building tools for users
  - *fermipy.diffuse* and *dmpipe* implement example analysis pipelines

RESOURCES

Package References

- ***fermitools*** (formerly ScienceTools): Fermi-LAT data analysis
  - https://github.com/fermi-lat/Fermitools-conda
- ***fermipy***: high level binned likelihood analysis of Fermi-LAT data
  - https://fermipy.readthedocs.io/
  - *fermipy.jobs*: tools to build analysis pipelines
  - *fermipy.diffuse*: tools for all-sky diffuse analysis
- ***dmpipe***: DM analysis pipeline
  - https://dmpipe.readthedocs.io/

fermitools package details:

- Installation:
  - conda create -n fermi -c conda-forge/label/cf201901 -c fermi fermitools
- Documentation: https://fermi.gsfc.nasa.gov/ssc/data/analysis/software
- Code repo: https://github.com/fermi-lat
- Maintainers:
  - Fermi Science Support Center
- Current version: fermitools 1.0.1
- Dependencies:
  - numpy, xml, cfitsio, healpix, astropy, wcslib, clhep, root ...

fermipy package details:

- Installation:
  - pip install fermipy
  - conda install fermipy
- Documentation: https://fermipy.readthedocs.io/
- Code repo: https://github.com/fermiPy/fermipy
- Python Package Index: https://pypi.org/project/fermipy/
- Developers:
  - Matthew Wood, EC, many others…
- Current version: fermipy 0.17.4
- Dependencies:
  - numpy, healpy, astropy, gammapy, fermitools

dmpipe package details:

- Installation:
  - pip install dmpipe
- Documentation: https://dmpipe.readthedocs.io/
- Code repo: https://github.com/fermiPy/dmpipe
- Python Package Index: https://pypi.org/project/dmpipe/
- Developers:
  - EC, Mattia di Mauro
- Current version: dmpipe 0.1.2
- Dependencies:
  - numpy, astropy, fermipy, dmsky

DM Pipeline Intermediate Data Products

- Target J factor maps
- Pre-prepared events, spacecraft and livetime cube files
- Target ROI analysis inputs
  - Counts maps, exposure maps, “source map” templates
  - Model definitions
- Target ROI baseline analysis
  - *fermipy* Region of interest “snapshots”
- Target SED analysis
  - *fermipy* SED likelihood FITS files, L(E, F_E)
    - https://gamma-astro-data-formats.readthedocs.io/en/latest/spectra/
- DM Likelihoods, L(m_χ, $\langle\sigma v\rangle$)
  - DM likelihood “castro” files, modified version of SED FITS files
- Simulation summary data
  - Expectation bands for limits and maximum likelihood estimate