Dark Matter flux from a dark Galactic subhalo

This tutorial demonstrates how to perform a simulation of the gamma-ray flux from dark matter annihilations in a dark Galactic subhalo in Fermi-LAT data. Before diving into the actual simulation of a Fermi-LAT measurement, we will study two fundamental ingredients needed for computing the gamma-ray flux:

1. the geometrical factor, i.e. the J factor

2. the injection spectrum dN/dE

The simulation is performed using fermipy, a python interface to the official Fermi Science Tools. We will then simulate the flux expected from a dark matter subhalo using the following data selection:

- 8x8 degree ROI
- Start Time (MET) = 239557417 seconds
- Stop Time (MET) = 620181124 seconds
- Minimum Energy = 500 MeV
- Maximum Energy = 1000e3 MeV
- zmax = 105 deg
- P8R3_SOURCE_V3 (evclass=128)

Import python and fermipy libraries

For an introduction to Fermipy, and the complete documentation: https://fermipy.readthedocs.io/en/latest/index.html

For the Fermi Science Tools: https://fermi.gsfc.nasa.gov/ssc/data/analysis/scitools/overview.html

In [2]:
```
%matplotlib inline
import os
import numpy as np
from fermipy.gtanalysis import GTAnalysis
from fermipy.plotting import ROIPlotter, SEDPlotter
import matplotlib.pyplot as plt
import matplotlib

if os.path.isfile('DMsubhalosim.tgz'):
    !tar -xvf DMsubhalosim.tgz
```

```
DMsubhalosim/
DMsubhalosim/ccube.fits
DMsubhalosim/ft1_00.fits
DMsubhalosim/data/
DMsubhalosim/data/P8R3_SOURCE_zmax105_gtselect_graspa23.fits
DMsubhalosim/data/gll_psc_v27.fit
DMsubhalosim/data/P8R3_SOURCE_zmax105_gtltcube.fits
DMsubhalosim/ccube_00.fits
DMsubhalosim/ccubemc_00.fits
DMsubhalosim/config.yaml
DMsubhalosim/clumpy/
DMsubhalosim/clumpy/annihil_gal2D_LOS180_0_FOVdiameter360.0deg_nside1024.drawn
DMsubhalosim/srcmap_00.fits
DMsubhalosim/bexpmap_roi_00.fits
DMsubhalosim/bexpmap_00.fits
```

The J factor from dark matter subhalos

The J factor is the integral along the line of sight of the dark matter density, squared since we consider dark matter annihilations.

The distribution of dark matter density in the Galaxy could be obtained with numerical simulations or with semi-analytical models for the clustering of dark matter structures.

Here we use the results of a simulation performed with the CLUMPY code (https://clumpy.gitlab.io/CLUMPY/, main developer: David Maurin). CLUMPY simulates dark matter subhalo populations and saves their properties (subhalo position in Galactic coordinates, J factor for an observed located at the Solar System position in the Galaxy, mass, distance from Solar System, ..) in a table.

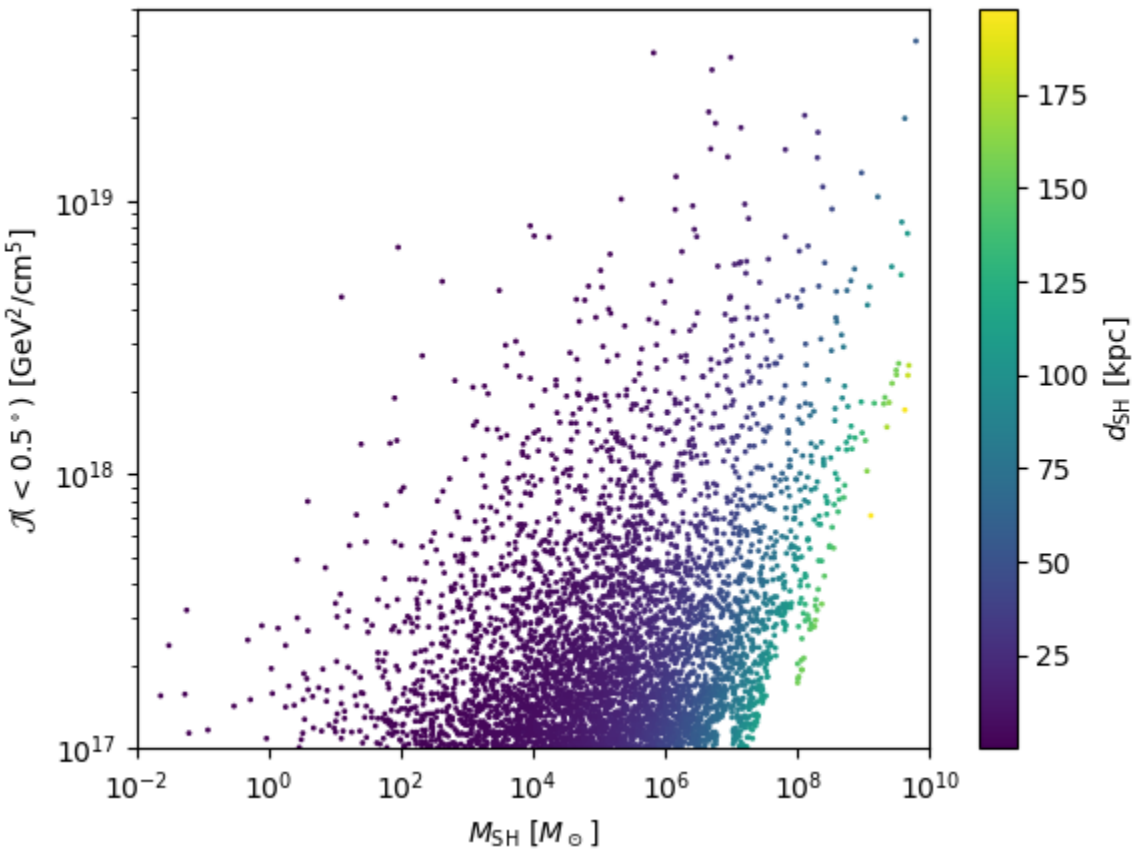
We will read this table and produce a similar plot to the Figure 1 from the following paper: https://arxiv.org/abs/1910.13722 to illustrate the properties of dark matter subhalos in a dark matter-only cosmological simulation.

OPTIONAL EXERCISES:

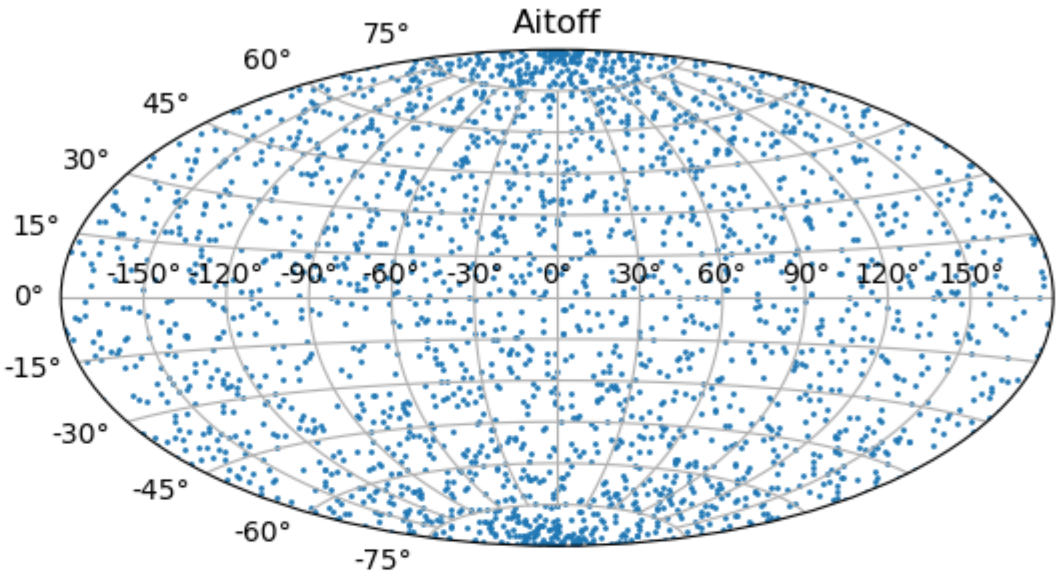
1. Produce a skymap with the positions of the dark matter subhalos in the simulation with J>1e17GeV^2/cm^5. You can choose among different sky projections available within matplotlib: https://matplotlib.org/3.1.1/gallery/subplots_axes_and_figures/geo_demo.html Pay attention to the coordinate system.

2. Produce an histogram for the dark matter subhalos masses, using both the units of solar masses and kg. What are the minimum, maximum values? How they compare to other objects in the Milky Way, apart for the Sun, or to the total dark matter mass of our Galaxy? Compare with Y.Genolini Astrophysics lecture.

In [3]:
```
#Reading the CLUMPY results; have also a visual look to the table file
J_file = 'DMsubhalosim/clumpy/annihil_gal2D_LOS180_0_FOVdiameter360.0deg_nside1024.drawn'
dat = np.genfromtxt(J_file, skip_header=3, skip_footer=1)

lons = dat[:,2]
lats = dat[:,3]
dists = dat[:,4]
J_factors = dat[:,13]
Masses = dat[:,15]

mask = np.where((J_factors > 1e17) & (J_factors <= 1e20) )[0]

cmap = plt.cm.viridis
plt.scatter(Masses[mask], J_factors[mask], s=1, c=dists[mask], cmap=cmap)
plt.ylim(1e17, 5e19)
plt.xlim(1e-2, 1e10)
plt.xscale('log')
plt.yscale('log')
plt.colorbar(label = r'$d_{\mathrm{SH}}$ [kpc]')
plt.xlabel(r'$M_{\mathrm{SH}}$ [$M_\odot$]')
plt.ylabel(r'$\mathcal{J}(<0.5^\circ)$ [GeV$^2$/cm$^5$]')
plt.show()
```



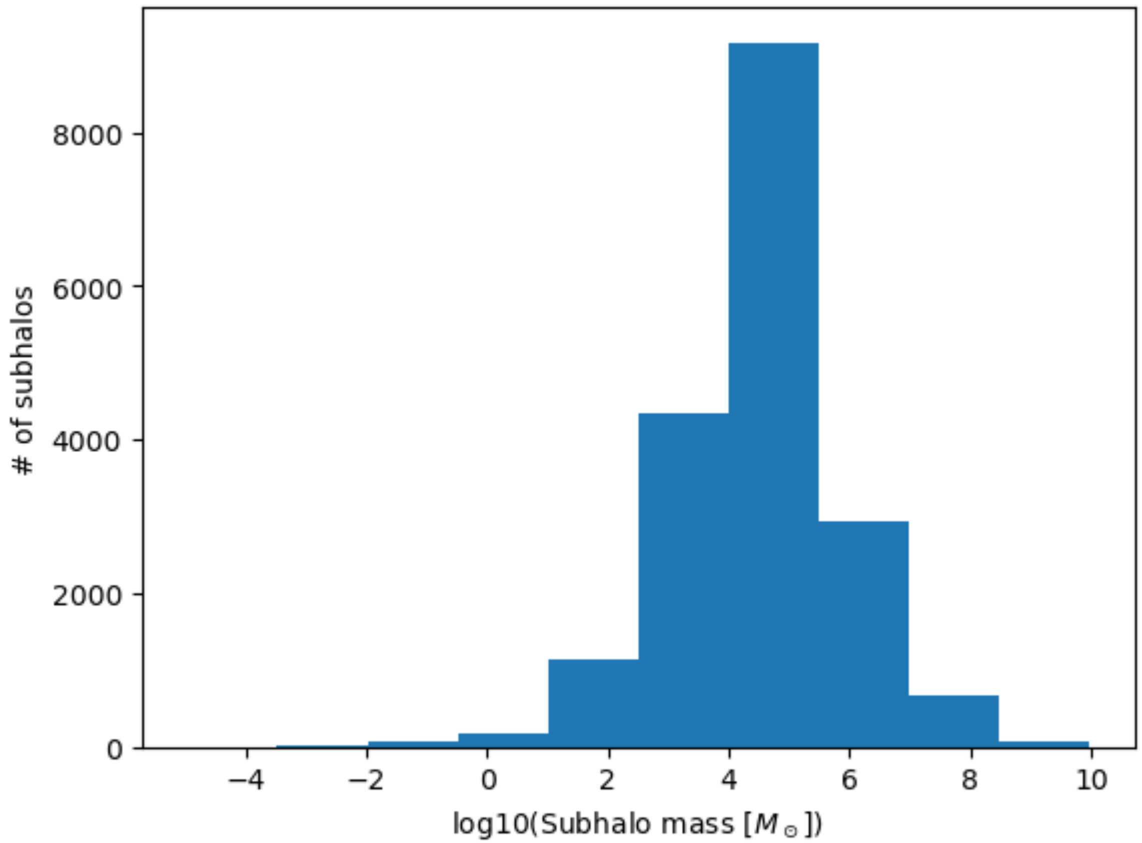
In [4]:
```
#Block for Optional points
#Skymap

plt.figure()
plt.subplot(111, projection="aitoff")
plt.scatter(lons[mask], lats[mask], s=1)
plt.title("Aitoff")
plt.grid(True)
```



In [ ]:

In [11]:

```
#Subhalo mass distribution
plt.hist(np.log10(Masses))
#lt.xscale('log')
#plt.yscale('log')
plt.xlabel(r'log10(Subhalo mass [$M_\odot$])')
plt.ylabel(r'# of subhalos')
#plt.title(' the dark matter subhalos masses')
plt.show()


print("Minimum mass:", min(Masses[mask]), "solar mass")
print("Maximum mass:", max(Masses[mask]), "solar mass")
```




```
Minimum mass: 0.00507 solar mass
Maximum mass: 6150000000.0 solar mass
```

The injection spectrum from dark matter annihilations

We now focus on the energy distribution dN/dE of gamma rays produced in dark matter annihilations. We will see how the spectrum changes depending on the annihilation channel and on the dark matter mass.

The energy distribution of final states in dark matter annihilations is computed by modeling the hadronization and/or decay of the annihilation products in frameworks such as DarkSUSY, which is based on Pythia (only Standard Model physics at this stage). Here we will use a 'DMFitFunction' as derived in this work: https://ui.adsabs.harvard.edu/abs/2008JCAP...11..003J/abstract and implemented in fermipy: https://fermipy.readthedocs.io/en/latest/_modules/fermipy/spectrum.html?highlight=dmfitfunction# which is a fit of results obtained with Monte Carlo simulations within DarkSUSY. More recent, broadly used repositories for the injection spectrum are available here:

-http://www.marcocirelli.net/PPPC4DMID.html

-https://github.com/nickrodd/HDMSpectra

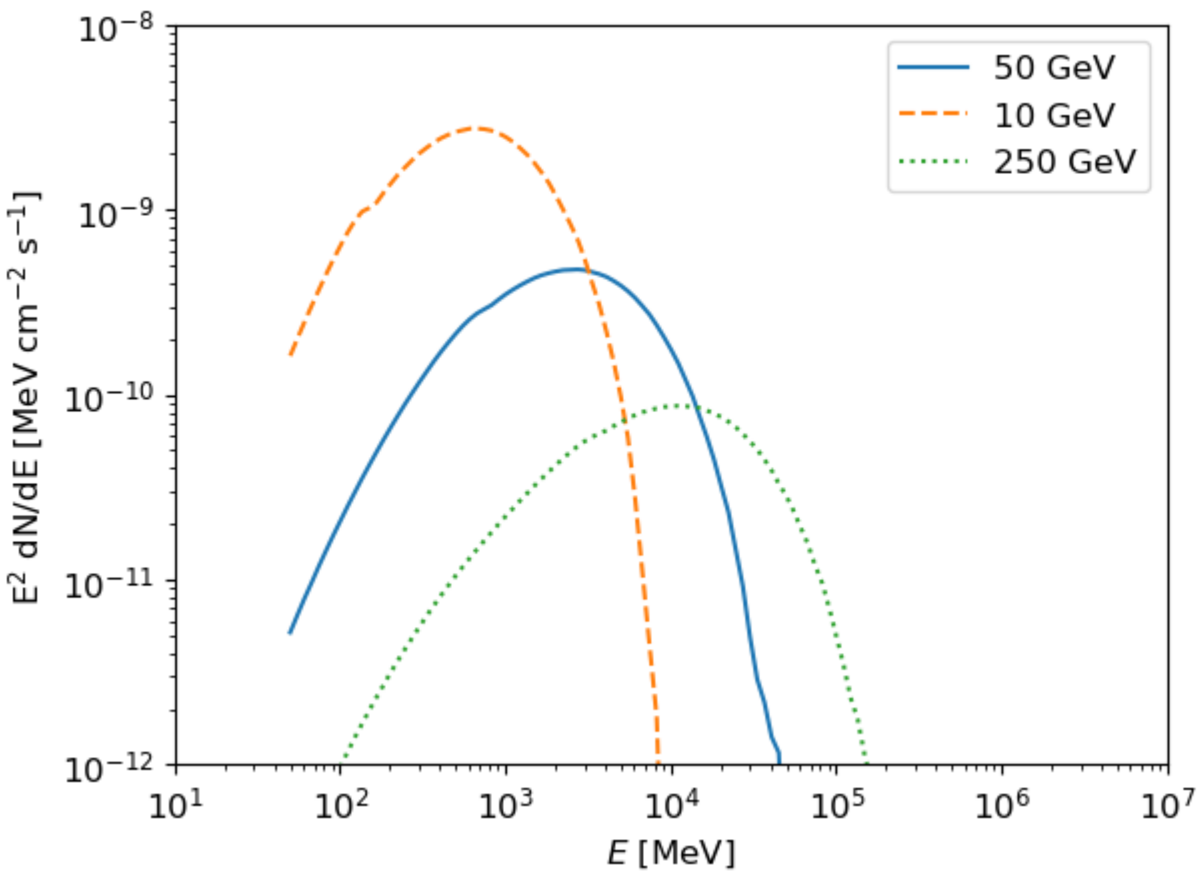
-https://github.com/ajueid/CosmiXs

The code below is not commented and provides an example on how to plot the dNdE (first block) and an optional exercise with the flux (second block). Your tasks, using the code documentation to understand how to use the functions, are:

0. Complete the axis labels with the correct units
1. Modify the code to show the dN/dE for the bb channel for 10 GeV, 50 GeV, 250 GeV for a thermal relic cross section of 3e-26 cm3/s
2. Modify the code to show the dN/dE for 50 GeV and at least two channels, for example bb and tau+ tau-.

Observe the results: can you explain the cutoff in the spectrum as a function of the mass? And the different shapes for different annihilation channels? (Please notice that we plot the spectrum dN/dE multiplied by E2)

In [26]:
```
from fermipy.spectrum import DMFitFunction

params = [3e-26, 50.0]
x = np.logspace(np.log10(50), np.log10(1000000), 100) # MeV
DMF_bb = DMFitFunction(params, chan = 'bb', jfactor = 1e+17)
dndE_bb = DMF_bb.e2dnde(x, params)

plt.ylim(1e-12, 1e-8)
plt.xlim(1e1, 1e7)

plt.xlabel(r'$E$ [MeV]')
plt.ylabel(r'E$^2$ dN/dE [MeV cm$^{-2}$ s$^{-1}$]')
plt.loglog(x, dndE_bb, label='50 GeV')


params = [3e-26, 10.0]
x = np.logspace(np.log10(50), np.log10(1000000), 100) # MeV
DMF_bb = DMFitFunction(params, chan = 'bb', jfactor = 1e+17)
dndE_bb = DMF_bb.e2dnde(x, params)
plt.loglog(x, dndE_bb, '--', label='10 GeV')

params = [3e-26, 250.0]
x = np.logspace(np.log10(50), np.log10(1000000), 100) # MeV
DMF_bb = DMFitFunction(params, chan = 'bb', jfactor = 1e+17)
dndE_bb = DMF_bb.e2dnde(x, params)
plt.loglog(x, dndE_bb, ':', label='250 GeV')
plt.legend()

plt.show()
```



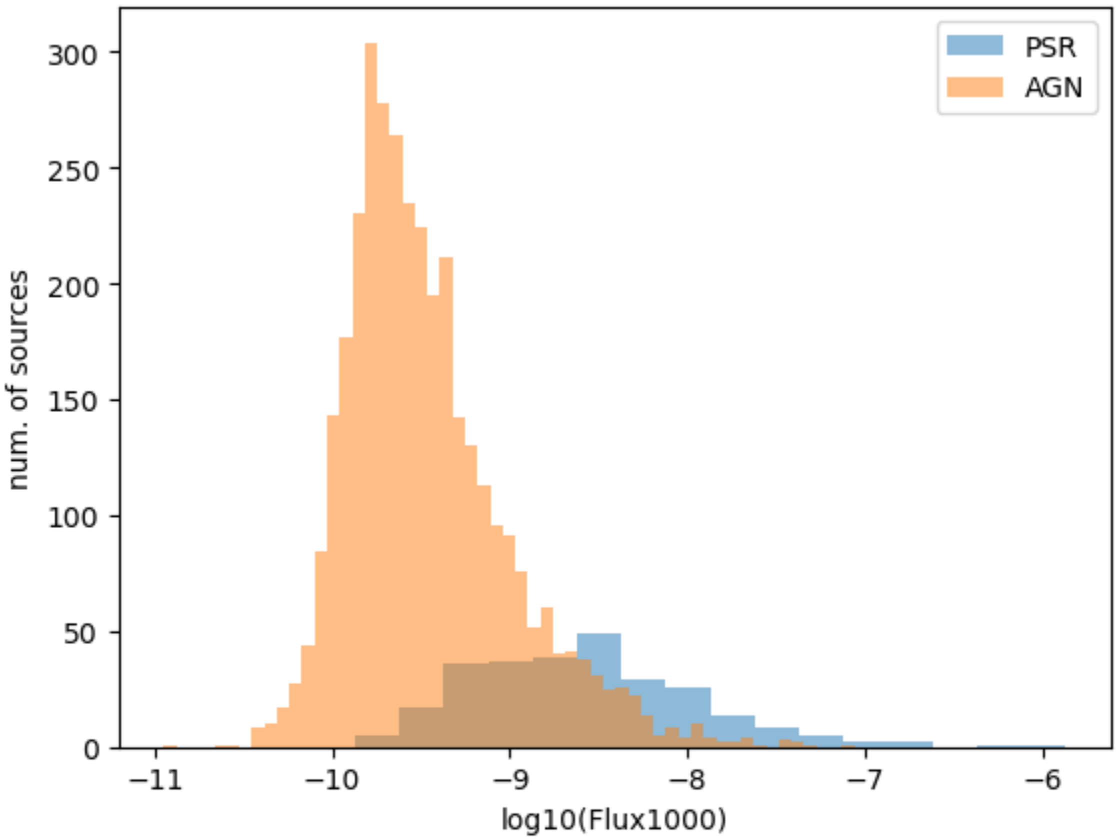
Optional : the integral flux from a dark matter subhalo compared to Fermi-LAT catalog sources

In [14]:
```
#Optional: compute the integral dark matter flux in an energy range
#and compare it with the flux from a source in the Fermi-LAT catalog.
#Choose different values for the J factor of the previous exercise, and
#different annihilation cross sections

params = [1e-26, 100.0]
x = np.logspace(np.log10(50), np.log10(1000000), 100) # MeV
DMF_bb = DMFitFunction(params, chan = 'bb', jfactor = 1e+19)
dndE_bb = DMF_bb.e2dnde(x, params)

Emin=1e3#MeV
Emax=1e5
flux = DMF_bb.flux(Emin, Emax, params)
print('Flux from DM subhalo', flux, '[ph cm-2 s-1]')
```

```
Flux from DM subhalo 5.433869867220774e-12 [ph cm-2 s-1]
```

In [15]:
```
#Read the catalog and the integral fluxes, compare with mean flux of all sources, and distribution for AGN and PSR.
from catalog import *
mycatalog=catalog('DMsubhalosim/data/gll_psc_v27.fit', 1)
mycatalog()
mycatalog.feat_stats('Flux1000') #Integral flux between 1-100 GeV units of ph cm-2 s-1
psrflux= mycatalog.sort_feat('Flux1000',mycatalog.class_psr)
agnflux= mycatalog.sort_feat('Flux1000',mycatalog.class_agn)
plt.hist(np.log10(psrflux), bins='auto', alpha=0.5, label='PSR')
plt.hist(np.log10(agnflux), bins='auto', alpha=0.5, label='AGN')
plt.xlabel('log10(Flux1000)')
plt.ylabel('num. of sources')
plt.legend()
plt.show()
```

```
This catalog has 5788 sources with 74 features
Stats Info for Flux1000
name = Flux1000
mean = 1.63641e-09
std = 2.08046e-08
min = 1.1304e-11
max = 1.35169e-06
n_bad = 0
length = 5788
```


PSR
AGN
num. of sources
log10(Flux1000)
0
50
100
150
200
250
300
−11
−10
−9
−8
−7
−6

Fermi-LAT measurement simulation

In this thread we will use a pregenerated data set which is contained in a tar archive to speed up the runtime. To run the fermipy simulation, we need:

1. A configuration file
2. A data file (called evfile) and a file containing the pointing history of Fermi-LAT (ltcube)
3. The details of the Fermi-LAT event selection, available within fermipy
4. Models for the backgrounds coming from cosmic rays and dim sources, available within fermipy
5. A catalog of gamma-ray sources already detected, available within fermipy; updated catalog is provided in the folder.

We are going to use a benchmark dark matter subhalo located aoutside of the Galactic plane.

This new source is added to the original model of the sky and then simulated.

We will begin by looking at the contents of the configuration file. Note: you need to change the path to the catalog file

In [16]:
```
!cat DMsubhalosim/config.yaml
```

```
data:
  evfile : 'data/P8R3_SOURCE_zmax105_gtselect_graspa23.fits'
  scfile : 'data/lat_spacecraft_merged.fits'
  ltcube : 'data/P8R3_SOURCE_zmax105_gtltcube.fits'

binning:
  roiwidth   : 8.0
  binsz      : 0.1 #
  binsperdec : 8
  coordsys   : 'GAL'

selection :
  emin : 50
  emax : 1000.e3 #MeV
  tmin : 239557417
  tmax : 620181124 # MET in s
  zmax    : 105
  evclass : 128
  evtype  : 3
  ra: 17.28
  dec: -1.79

# gtmktime parameters
  filter : 'DATA_QUAL>0 && LAT_CONFIG==1'

gtlike:
  edisp : True
  irfs : 'P8R3_SOURCE_V3'
  edisp_disable : ['isodiff','galdiff']

model:
  src_roiwidth : 8.0
  galdiff  : 'gll_iem_v07.fits'
  isodiff  : 'iso_P8R3_SOURCE_V3_v1.txt'
  catalogs : '/home/graspa/Astroparticle_exercise/DMsubhalosim/data/gll_psc_v27.fit'



fileio:
  usescratch: False
```

To get started we will first instantiate a GTAnalysis instance using the config file in the directory and the run the setup() method. This will prepare all the ancillary files and create the pylikelihood instance for binned analysis. Note that in this example these files have already been generated so the routines that would normally be executed to create these files will be skipped.

In [17]:
```
gta = GTAnalysis('DMsubhalosim/config.yaml')
matplotlib.interactive(True)
gta.setup()
gta.write_roi('setup')
```

```
2024-07-20 13:48:06 INFO    GTAnalysis.__init__():
--------------------------------------------------------------------------------
fermipy version 1.2.0
ScienceTools version 2.2.0
2024-07-20 13:48:07 INFO    GTAnalysis.setup(): Running setup.
2024-07-20 13:48:07 INFO    GTBinnedAnalysis.setup(): Running setup for component 00
2024-07-20 13:48:07 INFO    GTBinnedAnalysis._select_data(): Skipping data selection.
2024-07-20 13:48:07 INFO    GTBinnedAnalysis.setup(): Using external LT cube.
2024-07-20 13:48:09 INFO    GTBinnedAnalysis._create_expcube(): Skipping gtexpcube.
WARNING: FITSFixedWarning: 'datfix' made the change 'Set DATEREF to '2001-01-01T00:01:04.184' from MJDREF.
Set MJD-OBS to 54682.655283 from DATE-OBS.
Set MJD-END to 59088.005927 from DATE-END'. [astropy.wcs.wcs]
2024-07-20 13:48:09 INFO    GTBinnedAnalysis._create_srcmaps(): Skipping gtsrcmaps.
2024-07-20 13:48:09 INFO    GTBinnedAnalysis.setup(): Finished setup for component 00
2024-07-20 13:48:09 INFO    GTBinnedAnalysis._create_binned_analysis(): Creating BinnedAnalysis for component 00.
2024-07-20 13:48:28 INFO    GTAnalysis.setup(): Initializing source properties
2024-07-20 13:48:38 INFO    GTAnalysis.setup(): Finished setup.
2024-07-20 13:48:38 INFO    GTBinnedAnalysis.write_xml(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/s
etup_00.xml...
2024-07-20 13:48:38 INFO    GTAnalysis.write_fits(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/setu
p.fits...
WARNING: Format %s cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
WARNING: Format %f cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
WARNING: Format %s cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
2024-07-20 13:48:57 INFO    GTAnalysis.write_roi(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/setup.n
py...
```

We then proceed to the simulation. As a simple example, we simulate a ROI in which only the DM subhalo is added, together with the isotropic and Galactic backgrounds. With the gta.add_source() we add a source in the center of the ROI with the spectral properties of a dark matter subhalo, see previous exercises. We then simulate the ROI and visualize the model and the gamma-ray counts map of the simulation after a fit. Up to you: change the properties of the DM subhalo (J factor, annihilation cross section, dark matter mass, channel) and see how the significance/number of photons/flux of the simulated DM subhalo varies! Do you always detect something?

In [27]:
```
gta.delete_sources(exclude=['isodiff','galdiff'])
gta.print_roi()

gta.add_source('DMsubhalo',
                   dict(rad=17.28, dec=-1.79,
                   norm=dict(value=5., scale=1e19, max="1e5",min="1e5",free="0"), #J factor
                   sigmav=dict(value=10., scale=1e-26, max="5000",min="0",free="0"), #sigmav
                   mass=dict(value=100., scale=1, max="5000",min="1",free="0"),  #dark matter mass
                   bratio=dict(value=1., scale=1, max="1.0",min="0.0",free="0"), #branching ration
                   channel0=dict(value=4., scale=1, max="10",min="1",free="0"),  #channel
                   SpectrumType='DMFitFunction'),
                   free=True, init_source=True)

gta.simulate_roi(name=None, randomize=False, restore=False)
gta.print_roi()
sim_results = gta.fit()
gta.print_roi()
gta.write_roi('sim_result', make_plots=True)

#This is the integrated flux in the energy range defined in the config file
print('DM subhalo flux', gta.roi.sources[0]['flux'], '[ph/cm^2/s]')

from IPython.display import IFrame
IFrame("DMsubhalosim/sim_result_counts_map_1.699_6.000.png", width=600, height=300)
```

```
2024-07-20 13:52:26 INFO    GTAnalysis.delete_source(): Deleting source 4FGL J0108.1-0039
2024-07-20 13:52:26 INFO    GTAnalysis.delete_source(): Deleting source 4FGL J0115.1-0129
2024-07-20 13:52:26 INFO    GTAnalysis.delete_source(): Deleting source 4FGL J0112.1-0321
2024-07-20 13:52:27 INFO    GTAnalysis.delete_source(): Deleting source 4FGL J0101.0-0059
2024-07-20 13:52:27 INFO    GTAnalysis.delete_source(): Deleting source 4FGL J0059.3-0152
2024-07-20 13:52:27 INFO    GTAnalysis.delete_source(): Deleting source 4FGL J0059.2+0006
2024-07-20 13:52:27 INFO    GTAnalysis.delete_source(): Deleting source 4FGL J0108.6+0134
2024-07-20 13:52:27 INFO    GTAnalysis.delete_source(): Deleting source 4FGL J0125.7-0015
2024-07-20 13:52:27 INFO    GTAnalysis.print_roi():
name                SpatialModel   SpectrumType     offset        ts       npred
--------------------------------------------------------------------------------
isodiff             ConstantValue  FileFunction      -----       nan     59763.5
galdiff             MapCubeFunctio PowerLaw          -----       nan     54098.1

2024-07-20 13:52:27 INFO    GTAnalysis.add_source(): Adding source DMsubhalo
2024-07-20 13:52:29 INFO    GTAnalysis.simulate_roi(): Simulating ROI
2024-07-20 13:52:30 INFO    GTAnalysis.simulate_roi(): Finished
2024-07-20 13:52:30 INFO    GTAnalysis.print_roi():
name                SpatialModel   SpectrumType     offset        ts       npred
--------------------------------------------------------------------------------
DMsubhalo           PointSource    DMFitFunction     0.000       nan       223.8
isodiff             ConstantValue  FileFunction      -----       nan     59763.5
galdiff             MapCubeFunctio PowerLaw          -----       nan     54098.1

2024-07-20 13:52:30 INFO    GTAnalysis.fit(): Starting fit.
2024-07-20 13:52:31 INFO    GTAnalysis.fit(): Fit returned successfully. Quality:   3 Status:   0
2024-07-20 13:52:31 INFO    GTAnalysis.fit(): LogLike:   -77339.971 DeltaLogLike:        0.000
2024-07-20 13:52:31 INFO    GTAnalysis.print_roi():
name                SpatialModel   SpectrumType     offset        ts       npred
--------------------------------------------------------------------------------
DMsubhalo           PointSource    DMFitFunction     0.000    123.99       223.8
isodiff             ConstantValue  FileFunction      -----       nan     59763.5
galdiff             MapCubeFunctio PowerLaw          -----       nan     54098.1

2024-07-20 13:52:31 INFO    GTBinnedAnalysis.write_xml(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/s
im_result_00.xml...
2024-07-20 13:52:31 INFO    GTAnalysis.write_fits(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/sim_re
sult.fits...
WARNING: Format %s cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
```

```
WARNING: Format %f cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn keyword. [astropy.io.fits.column]
WARNING: Format %s cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn keyword. [astropy.io.fits.column]
2024-07-20 13:52:47 INFO    GTAnalysis.write_roi(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/sim_result.npy...
DM subhalo flux 6.890020799637481e-10 [ph/cm^2/s]
```

Out[27]:

Firefox Can’t Open This Page

To protect your security, localhost will not allow Firefox to display the page if another site has embedded it. To see this page, you need to open it in a new window.

Learn more…

Optional: a view to the gamma-ray sky model from the catalog

In the simulation above, before simulating the gamma-ray flux from the DM subhalo we have deleted all the point sources from the ROI model. In what follows we will simulate a model for the gamma-ray sky in this ROI by taking the sources available in the Fermi-LAT catalog plus the DM subhalo.

First, we load the ROI, and add the DM subhalo again. We then see a list of sources in the model of the gamma-ray sky along with their distance from the ROI center (offset), TS, and number of predicted counts (Npred). Since we haven't yet fit any sources, the significance (TS) of all sources included in the model will initially be assigned as nan. The model contains the sources as found in the Fermi-LAT catalog, and diffuse and isotropic emissions.

In [28]:

```
gta.load_roi('setup')
gta.add_source('DMsubhalo',
                   dict(rad=17.28, dec=-1.79,
                   norm=dict(value=5., scale=1e19, max="1e5",min="1e5",free="0"), #J factor
                   sigmav=dict(value=10., scale=1e-26, max="5000",min="0",free="0"), #sigmav
                   mass=dict(value=100., scale=1, max="5000",min="1",free="0"),  #dark matter mass
                   bratio=dict(value=1., scale=1, max="1.0",min="0.0",free="0"), #branching ratio
                   channel0=dict(value=4., scale=1, max="10",min="1",free="0"),  #channel
                   SpectrumType='DMFitFunction'),
                   free=True, init_source=True)
gta.print_roi()
```

```
2024-07-20 13:52:58 INFO    GTAnalysis.load_roi(): Loading ROI file: /home/graspa/Astroparticle_exercise/DMsubhalosi
m/setup.npy
2024-07-20 13:52:58 INFO    GTBinnedAnalysis._create_binned_analysis(): Creating BinnedAnalysis for component 00.
2024-07-20 13:53:17 INFO    GTAnalysis.load_roi(): Finished Loading ROI
2024-07-20 13:53:17 INFO    GTAnalysis.add_source(): Adding source DMsubhalo
2024-07-20 13:53:19 INFO    GTAnalysis.print_roi():
name                SpatialModel   SpectrumType     offset        ts       npred
--------------------------------------------------------------------------------
DMsubhalo           PointSource    DMFitFunction     0.000       nan       223.8
4FGL J0108.1-0039   PointSource    PowerLaw          1.159       nan      2482.3
4FGL J0115.1-0129   PointSource    PowerLaw          1.534       nan      2882.7
4FGL J0112.1-0321   PointSource    PowerLaw          1.742       nan      1836.0
4FGL J0101.0-0059   PointSource    PowerLaw          2.169       nan       800.9
4FGL J0059.3-0152   PointSource    PowerLaw          2.444       nan       351.3
4FGL J0059.2+0006   PointSource    PowerLaw          3.122       nan       789.3
4FGL J0108.6+0134   PointSource    LogParabola       3.374       nan     41375.3
4FGL J0125.7-0015   PointSource    PowerLaw          4.418       nan      1697.7
isodiff             ConstantValue  FileFunction      -----       nan     59763.5
galdiff             MapCubeFunctio PowerLaw          -----       nan     54098.1
```

Now we will run the *optimize* method. This method will iteratively optimize the parameters of all components in the ROI in several stages:

- Simultaneously fitting the normalization of the brightest model components containing at least some fraction of the total model counts (default 95%).
- Individually fitting the normalization of all remaining sources if they have Npred above some threshold (default 1).
- Individually fitting the normalization and shape of any component with TS larger than some threshold (default 25).

Running *optimize* gives us a baseline model that we can use as a starting point for subsequent stages of the analysis. We will also save the results of the analysis with write_roi. By saving the analysis state we can restore the analysis to this point at any time with the *load_roi* method.

In [29]:
```
gta.optimize()
gta.write_roi('optimize')
```

```
2024-07-20 13:53:28 INFO    GTAnalysis.optimize(): Starting
Joint fit  ['isodiff', 'galdiff', '4FGL J0108.6+0134', '4FGL J0115.1-0129']
Fitting shape galdiff TS:   3126.609
Fitting shape isodiff TS:   2041.758
Fitting shape DMsubhalo TS:    134.522
Fitting shape 4FGL J0125.7-0015 TS:     82.379
Fitting shape 4FGL J0112.1-0321 TS:     38.213
2024-07-20 13:53:41 INFO    GTAnalysis.optimize(): Finished
2024-07-20 13:53:41 INFO    GTAnalysis.optimize(): LogLike: -77372.243134 Delta-LogLike: 18925.669335
2024-07-20 13:53:41 INFO    GTAnalysis.optimize(): Execution time: 13.26 s
2024-07-20 13:53:41 INFO    GTBinnedAnalysis.write_xml(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/o
ptimize_00.xml...
2024-07-20 13:53:41 INFO    GTAnalysis.write_fits(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/optimi
ze.fits...
WARNING: Format %s cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
WARNING: Format %f cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
WARNING: Format %s cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
2024-07-20 13:53:58 INFO    GTAnalysis.write_roi(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/optimiz
e.npy...
```

After running *optimize* we can rerun *print_roi* to see a summary of the updated model. All sources that were fit in this step now have ts values and an Npred value the reflects the optimized normalization of that source. Note that model components that were not fit during the optimize step still have ts=nan.

In [30]:
```
gta.print_roi()
```

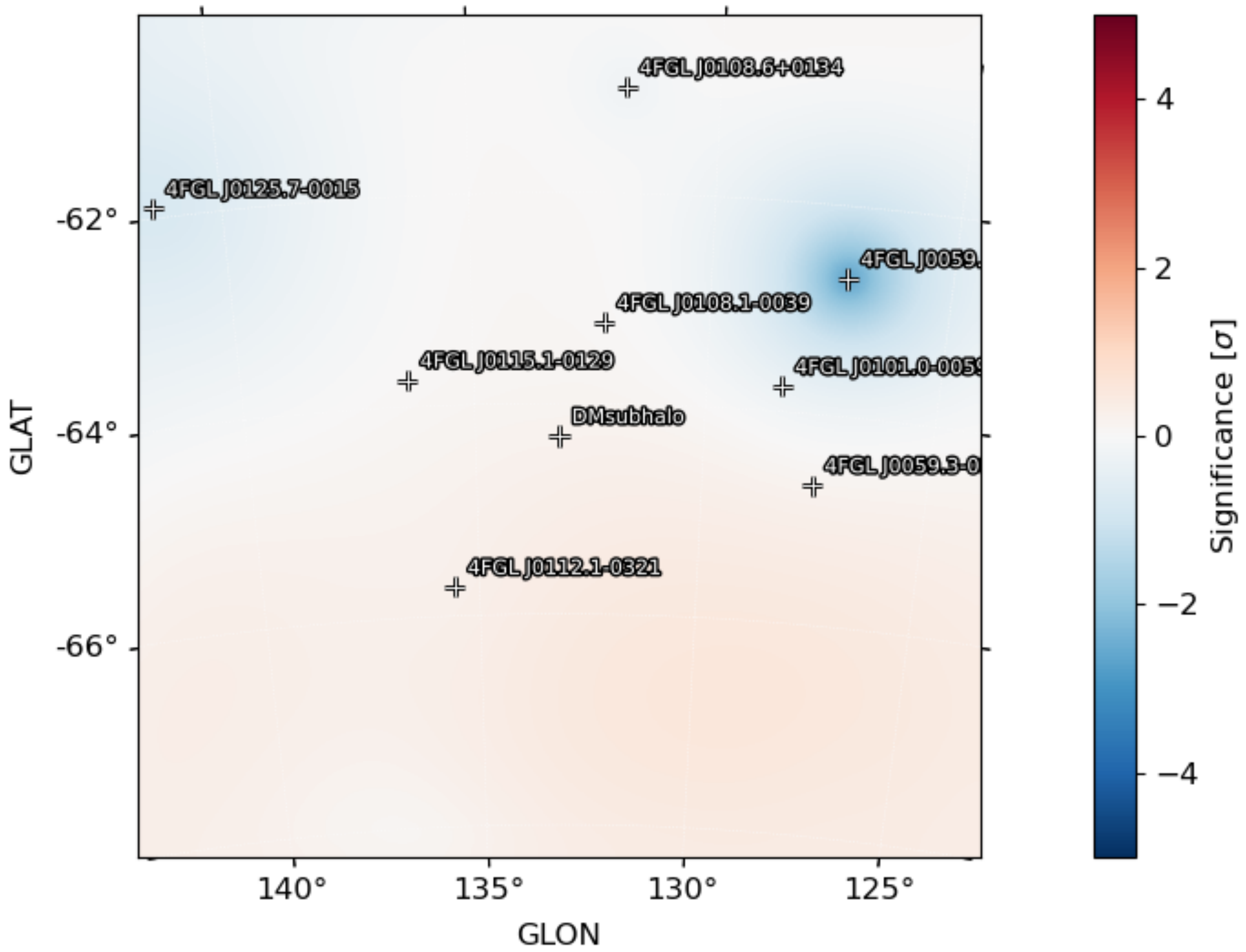
```
2024-07-20 13:54:07 INFO    GTAnalysis.print_roi():
name                SpatialModel   SpectrumType     offset        ts       npred
--------------------------------------------------------------------------------
DMsubhalo           PointSource    DMFitFunction     0.000    124.99       209.0
4FGL J0108.1-0039   PointSource    PowerLaw          1.159     15.93       745.8
4FGL J0115.1-0129   PointSource    PowerLaw          1.534     -0.00         0.0
4FGL J0112.1-0321   PointSource    PowerLaw          1.742     15.07       801.3
4FGL J0101.0-0059   PointSource    PowerLaw          2.169      5.36       276.9
4FGL J0059.3-0152   PointSource    PowerLaw          2.444      0.36        14.6
4FGL J0059.2+0006   PointSource    PowerLaw          3.122     18.60       500.0
4FGL J0108.6+0134   PointSource    LogParabola       3.374      0.45        52.3
4FGL J0125.7-0015   PointSource    PowerLaw          4.418     22.17       766.3
isodiff             ConstantValue  FileFunction      -----  30850.59     48460.4
galdiff             MapCubeFunctio PowerLaw          -----  57451.72     61565.6
```

We finally run a fit to further optimize the model of the sky. We free the parameters of sources at maximum 3 degrees of distance from the center of the ROI. To evaluate the quality of the optimized model we produce a residual map.

In [31]:
```
gta.free_sources(distance=3.0,pars='norm')
gta.free_sources(distance=3.0,pars='shape',minmax_ts=[100.,None])
fit_results = gta.fit()
gta.print_roi()
resid = gta.residmap('roi_postfit',model={'SpatialModel' : 'PointSource', 'Index' : 2.0}, make_plots=True)
fig = plt.figure(figsize=(14,6))
ROIPlotter(resid['sigma'],roi=gta.roi).plot(vmin=-5,vmax=5,levels=[-5,-3,3,5],subplot=121,cmap='RdBu_r')
plt.gca().set_title('Significance')

from IPython import display
display.Image('DMsubhalosim/roi_postfit_pointsource_powerlaw_2.00_residmap_sigma.png')
#IFrame("DMsubhalosim/roi_postfit_pointsource_powerlaw_2.00_residmap_sigma.png", width=600, height=300)
```

```
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for 4FGL J0108.1-0039     : ['Prefactor']
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for 4FGL J0115.1-0129     : ['Prefactor']
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for 4FGL J0112.1-0321     : ['Prefactor']
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for 4FGL J0101.0-0059     : ['Prefactor']
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for 4FGL J0059.3-0152     : ['Prefactor']
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for isodiff               : ['Normalizatio
n']
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for galdiff               : ['Prefactor']
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for DMsubhalo             : ['mass']
2024-07-20 13:54:10 INFO    GTAnalysis.free_source(): Freeing parameters for galdiff               : ['Index']
2024-07-20 13:54:10 INFO    GTAnalysis.fit(): Starting fit.
2024-07-20 13:54:36 INFO    GTAnalysis.fit(): Fit returned successfully. Quality:   3 Status:   0
2024-07-20 13:54:36 INFO    GTAnalysis.fit(): LogLike:   -77354.495 DeltaLogLike:       17.748
2024-07-20 13:54:36 INFO    GTAnalysis.print_roi():
name                SpatialModel   SpectrumType     offset        ts       npred
--------------------------------------------------------------------------------
DMsubhalo           PointSource    DMFitFunction     0.000    124.37       224.3
4FGL J0108.1-0039   PointSource    PowerLaw          1.159     -0.00         0.3
4FGL J0115.1-0129   PointSource    PowerLaw          1.534     -0.00         0.0
4FGL J0112.1-0321   PointSource    PowerLaw          1.742      0.22       121.6
4FGL J0101.0-0059   PointSource    PowerLaw          2.169     -0.00         0.0
4FGL J0059.3-0152   PointSource    PowerLaw          2.444      0.00         0.0
4FGL J0059.2+0006   PointSource    PowerLaw          3.122     18.60       500.0
4FGL J0108.6+0134   PointSource    LogParabola       3.374      0.45        52.3
4FGL J0125.7-0015   PointSource    PowerLaw          4.418     22.17       766.3
isodiff             ConstantValue  FileFunction      -----   2666.25     55583.9
galdiff             MapCubeFunctio PowerLaw          -----   2887.03     56863.3

2024-07-20 13:54:36 INFO    GTAnalysis.residmap(): Generating residual maps
2024-07-20 13:54:36 INFO    GTAnalysis.add_source(): Adding source residmap_testsource
2024-07-20 13:54:38 INFO    GTAnalysis.delete_source(): Deleting source residmap_testsource
2024-07-20 13:54:42 INFO    GTAnalysis.residmap(): Finished residual maps
2024-07-20 13:54:55 WARNING GTAnalysis.residmap(): Saving maps in .npy files is disabled b/c of incompatibilities in
python3, remove the maps from the /home/graspa/Astroparticle_exercise/DMsubhalosim/roi_postfit_pointsource_powerlaw_
2.00_residmap.npy
2024-07-20 13:54:55 INFO    GTAnalysis.residmap(): Execution time: 19.29 s
```

Out[31]:



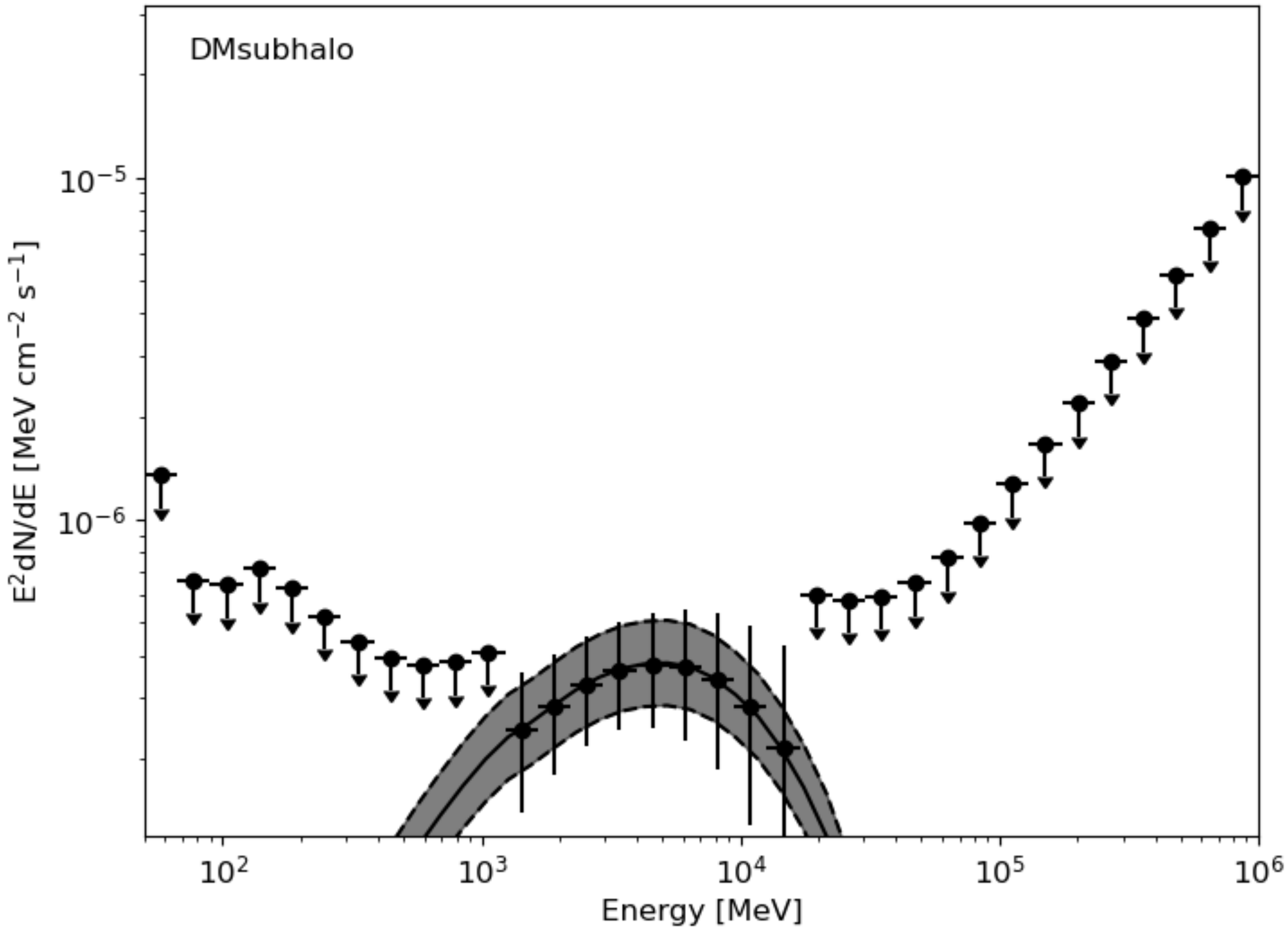
With the fitted model we can for example evaluate the spectral energy distribution (SED) of the sources within the ROI. We consider for example the DM subhalo source.

In [32]:
```
sed_source = gta.sed('DMsubhalo', make_plots=True)
gta.write_roi('fit_sed')
```

```
2024-07-20 13:55:02 INFO    GTAnalysis.sed(): Computing SED for DMsubhalo
2024-07-20 13:55:08 INFO    GTAnalysis._make_sed(): Fitting SED
2024-07-20 13:55:08 INFO    GTAnalysis.free_source(): Fixing parameters for DMsubhalo             : ['mass']
2024-07-20 13:55:08 INFO    GTAnalysis.free_source(): Fixing parameters for galdiff               : ['Index']
2024-07-20 13:55:18 INFO    GTAnalysis.sed(): Finished SED
2024-07-20 13:55:30 INFO    GTAnalysis.sed(): Execution time: 27.73 s
2024-07-20 13:55:30 INFO    GTBinnedAnalysis.write_xml(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/f
it_sed_00.xml...
2024-07-20 13:55:30 INFO    GTAnalysis.write_fits(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/fit_se
d.fits...
WARNING: Format %s cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
WARNING: Format %f cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
WARNING: Format %s cannot be mapped to the accepted TDISPn keyword values.  Format will not be moved into TDISPn key
word. [astropy.io.fits.column]
2024-07-20 13:55:46 INFO    GTAnalysis.write_roi(): Writing /home/graspa/Astroparticle_exercise/DMsubhalosim/fit_se
d.npy...
```

In [33]:
```
display.Image("DMsubhalosim/dmsubhalo_sed.png")
```

Out[33]:


DMsubhalo
10^{-5}
10^{-6}
E^2dN/dE [MeV cm^{-2} s^{-1}]
10^2
10^3
10^4
10^5
10^6
Energy [MeV]

In [ ]: