3ML
The Multi-Mission Maximum Likelihood framework

- Not your usual spectral fitting package
- Multi-wavelength, multi-messenger architecture
- Use plugins: you can interact with heterogeneous data in a unified way, while they are handled differently behind the scene
- Maximum Likelihood and Bayesian analysis
- Local and global minimization
- Simulations capabilities
- Python-based

https://github.com/giacomov/3ML

G. Vianello (Stanford University),
the 3ML team
3ML is different

- Existing solutions (xspec, sherpa...) are more or less a one-size-fits-all approach
- Only good for instruments which can be reconducted to the same analysis

* Plugin: connects 3ML with the instrument-specific software
  * Receive in input the model, give in output the likelihood value
  * No constraints on:
    * Messenger, data formats, likelihood function, background estimation and handling, language (C++, fortran...)
  * Existing solutions (ST, sherpa, xspec, gammapy, isis) can be plugins

**3ML architecture**

- Model definition + initial parameter values
- Model evaluation
- Optimizer selection configuration
- Optimization
- Model evaluation
- Likelihood
- Model
- Plugin
A MW analysis with 3ML:

```python
from threeML import *

# Download data
tstart = '2010-01-01 00:00:00'
tstop = '2010-01-08 00:00:00'
evfile, scfile = download_LAT_data(ra, dec, 20.0, tstart, tstop, time_type='Gregorian',
    destination_directory='Crab_data')

# Interrogate the public catalog (currently 3FGL)
lat_catalog = FermiLATSourceCatalog()
ra, dec, table = lat_catalog.search_around_source('Crab', radius=20.0)

# Get likelihood model from the table
model = lat_catalog.get_model()
model.free_point_sources_within_radius(3.0, normalization_only=True)

# Use Fermipy to analyze the data with the model
config = FermipyLike.get_basic_config(evfile=evfile, scfile=scfile, ra=ra, dec=dec)
LAT = FermipyLike('LAT', config)

veritas = VERITASLike('VERITAS', ...

hawc = HAWCLike('HAWC', ...)

ground = PhotometryLike('GROUND', filters=threeML_filter_library.ESO.GROND,  
i=(21.0, 0.1), z=(21.2, 0.1),  
J=(19.6, 0.1), H=(18.0, 0.1), K=(18.0, 0.1))
data = Dataset(LAT, hawc, veritas, ground)

j1 = Jointlikelihood(model, data)
best_fit_parameters, likelihood_values = j1.fit()
```

- Available plugins: Fermi/LAT, HAWC, VERITAS, OGIP-like (Swift, Chandra, XMM, Fermi/GBM, Konus...), Optical telescopes (2000 filters), SED data
- More in development (polarization)
- Extended source support is in development (fully available only for HAWC at the moment)

A posterior probability visualization for a Bayesian analysis

A contour plot from a Maximum Likelihood analysis
Join the 3ML team!

- 3ML is currently an effort of ~10 people
- Development is following the scientific interests of these people
- Join the team! We need people who are willing to contribute
- Repositories:
  - https://github.com/giacomov/3ML
  - https://github.com/giacomov/astromodels
- Docs (under development):
  - threeml.readthedocs.org
  - astromodels.readthedocs.org