CTA-GPS with gammapy version almost 1.0

Quentin Remy

XML -> gammapy models -> YAML and FITS

```
</source>
<source name="LS 5039" type="PointSource">
  <spectrum type="ExponentialCutoffPowerLaw">
    <parameter name="Prefactor" value="2.48" error="0" scale="1e-18" min="1e-07" max="1000" free="1" />
    <parameter name="Index" value="2.2" error="0" scale="-1" min="0" max="5" free="1" />
    <parameter name="CutoffEnergy" value="6.6" error="0" scale="1000000" min="0.01" max="1000" free="1" />
    <parameter name="PivotEnergy" value="1" scale="1000000" min="0.01" max="1000" free="0" />
  </spectrum>
  <spatialModel type="PointSource">
    <parameter name="RA" value="276.5625" error="0" scale="1" min="-360" max="360" free="1" />
    <parameter name="DEC" value="-14.825" error="0" scale="1" min="-90" max="90" free="1" />
  </spatialModel>
  <temporal type="PhaseCurve" file="phasecurve_LS_DC.fits" normalize="0">
    <parameter name="Normalization" value="1" scale="1" min="0" max="1000" free="0" />
    <parameter name="MJD" value="51943.09" scale="1" min="0" max="100000" free="0" />
    <parameter name="Phase" value="0" scale="1" min="0" max="1" free="0" />
    <parameter name="F0" value="2.967711" scale="1e-06" min="0" max="1000" free="0" />
    <parameter name="F1" value="0" scale="1" min="0" max="1000" free="0" />
    <parameter name="F2" value="0" scale="1" min="0" max="1000" free="0" />
  </temporal>
</source>
```

```
name: LS 5039
type: SkyModel
spectral:
    type: ExpCutoffPowerLawSpectralModel
    parameters:
    - {name: index, value: 2.2, unit: '', min: 0.0, max: 5.0, frozen: false, error: 0}
    - {name: amplitude, value: 1.5996e-18, unit: cm-2 MeV-1 s-1, min: 1.0e-25,
       max: 1.0e-15, frozen: false, error: 0}
    - {name: reference, value: 1000000.0, unit: MeV, min: 10000.0, max: 1000000000.0,
       frozen: true, error: 0}
    - {name: lambda_, value: 1.5151515151515152e-07, unit: MeV-1, min: 0.0001,
       max: 1.0e-09, frozen: false, error: 0}
    - {name: alpha, value: 1.0, unit: '', min: .nan, max: .nan, frozen: true,
       error: 0}
spatial:
    type: PointSpatialModel
    frame: icrs
    parameters:
    - {name: lon_0, value: 276.5625, unit: deg, min: -360.0, max: 360.0, frozen: false,
       error: 0}
    - {name: lat_0, value: -14.825, unit: deg, min: -90.0, max: 90.0, frozen: false,
       error: 0}
```

Missing models

- Broken-power-law (given as SmoothBrokenPowerLawSpectralModel with beta = 0.1 for now
- Piece-wise broken power law (ctool NodeFunction given as TemplateSpectralModel for now)
- Phase curve (returning mean spectrum instead)

Difference of conventions fixed case-by-case

Several missing unit in templates assumed to be default, user must check

Input/Output catalogue as FITS: in gammapy.catalogue ?

Source_Name	Npred	TS	SpatialModel	GLON	GLAT	Radius	Width	SpectralModel	F_0.05-1TeV	F_1-100TeV	TemporalModel
HESS J1640-465	5858.354...	10740.49...	GaussianSpatialModel	338.2771...	-0.00351...	0.071666...		ExpCutoffPowerLawSpe	9.179144944138073E-11	2.248657423459255E-12	
RCW 86	4959.837...	10545.96...	ShellSpatialModel	315.0927...	-2.37651...	0.151999...	0.037999...	ExpCutoffPowerLawSpe	3.105955076465605E-11	2.443045548847225E-12	
Galactic Centre	5030.374...	10403.79...	PointSpatialModel	359.9442...	-0.04616...	0.0		ExpCutoffPowerLawSpe	6.493502957560793E-11	1.703835602848676E-12	
composite_89	3439.444...	10337.65...	GaussianSpatialModel	51.24686...	0.543028...	0.029659...		LogParabolaSpectralM	1.1281786269079497E-11	1.0958932329316329E-12	
pwn_134	2450.918...	8403.256...	GaussianSpatialModel	50.47818...	1.842198...	0.009999...		LogParabolaSpectralM	2.935243020862899E-11	1.0846108082843142E-12	
2FHL_J0035.8+5949	1901.501...	8081.818...	PointSpatialModel	120.9719...	-2.98122...	0.0		PowerLawSpectralMode	1.2318021602730767E-10	3.1608730390042528E-12	
HESS J1708-443	6896.980...	7631.951...	GaussianSpatialModel	343.0646...	-2.32954...	0.289999...		PowerLawSpectralMode	7.979999994134346E-11	4.158000332932232E-12	
MGRO J1908+06	10344.95...	7549.803...	GaussianSpatialModel	40.40084...	-1.00098...	0.340000...		PowerLawSpectralMode	9.78005454399522E-11	3.7398899081775294E-12	
composite_304	4555.500...	7107.78125	GaussianSpatialModel	55.89612...	0.122399...	0.087410...		LogParabolaSpectralM	1.3269509623048403E-11	1.4000274464048834E-12	
HESS J1834-087	4964.926...	6765.946...	GaussianSpatialModel	23.26833...	-0.32988...	0.090000...		PowerLaw2SpectralMod	1.3776900453787988E-10	1.810456081927092E-12	
HESS J1731-347	5765.275...	6718.845...	ShellSpatialModel	353.5650...	-0.62219...	0.216000...	0.054000...	PowerLawSpectralMode	1.0261790722720932E-10	2.0011249601825654E-12	

GPS catalog production attempt in short

Datasets	Energies (TeV)	spectral margin	Offset max	binning	Spatial margin
SubTeV	0.05 – 1 10 bins/decade	2 extra bins	2.5°	0.06°	Sub-regions: 4° wider than from mask-fit 1° erosion of mask-fit from mask-safe
PosTeV	1 – 100 10 bins/decade	same	3.5°	0.03°	same

- Backgrounds:
  - irf: FoV-bkg (exclusion region from HGPS significance maps filtered by hysteresis initially)
  - diffuse: Fermi-LAT extrapolated

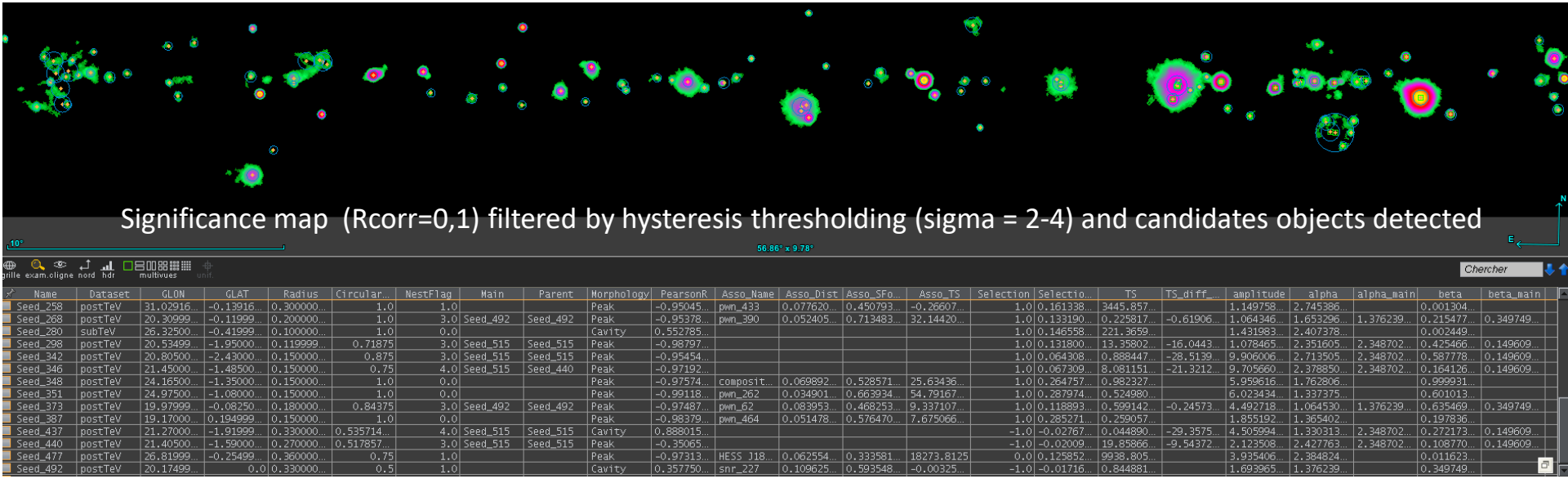
- Significance with Rcorr={0.1°,0.2°} on each dataset used for seed detection

- Joint fit of the two datasets, modelling seeds as 2D Sersic + LogParabola
(generic enough for classification then more specific models can be tested depending on the previous results)

Object Detection

- Filtered significance maps (gammapy + scikit-image)
  - -> Peak detection (gammapy)
  - -> Hough circle detection (scikit-image)
- Known source and simulated sources association using surface overlap criterion
- Seeds classification and ranking for selection (scikit-learn)
- Seeds properties: additional information to help on model selection and initial parameters setup

  gammapy.catalog could include a seed catalog class with some of these functionality, in particular nesting information can be very useful



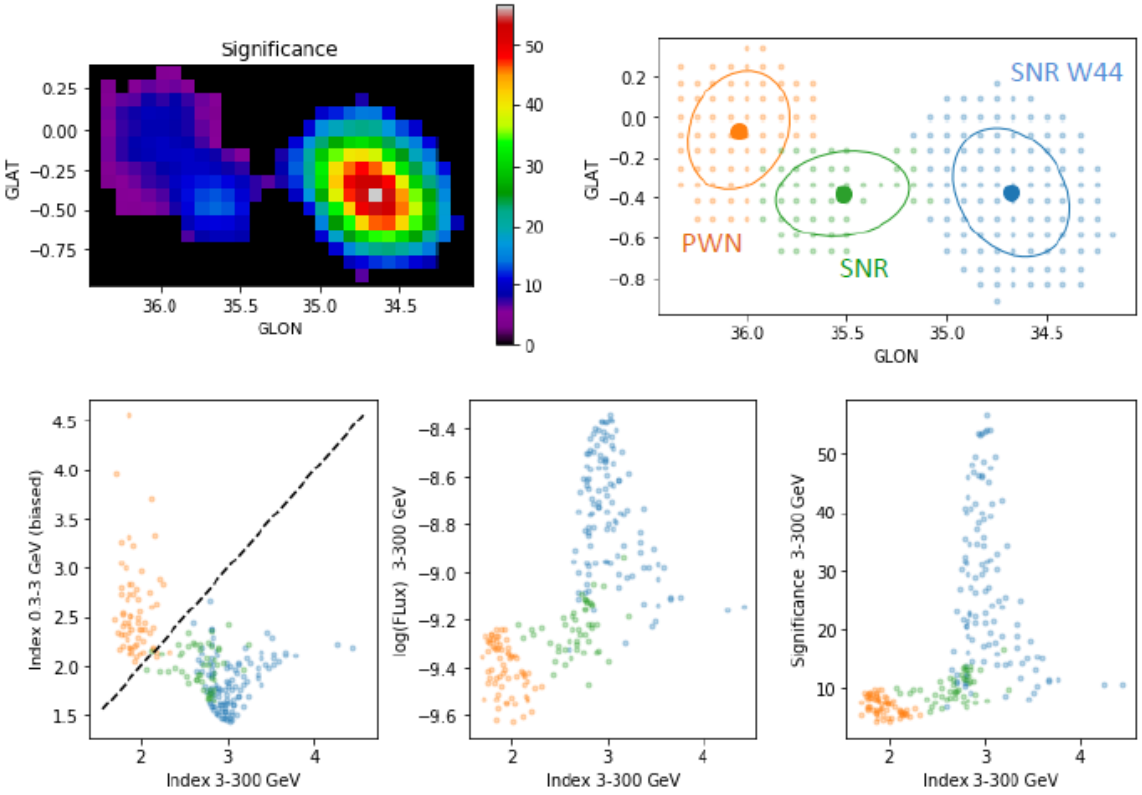
Significance map (Rcorr=0,1) filtered by hysteresis thresholding (sigma = 2-4) and candidates objects detected

Name	Dataset	GLON	GLAT	Radius	Circular...	NestFlag	Main	Parent	Morphology	PearsonR	Asso_Name	Asso_Dist	Asso_SFo...	Asso_TS	Selection	Selectio...	TS	TS_diff_...	amplitude	alpha	alpha_main	beta	beta_main
Seed_258	postTeV	31.02916...	-0.13916...	0.300000...	1.0	1.0			Peak	-0.95045...	pwn_433	0.077620...	0.450793...	-0.26607...	1.0	0.161338...	3445.857...		1.149758...	2.745386...		0.001304...	
Seed_268	postTeV	20.30999...	-0.11999...	0.200000...	1.0	3.0	Seed_492	Seed_492	Peak	-0.95378...	pwn_390	0.052405...	0.713483...	32.14420...	1.0	0.133190...	0.225817...	-0.61906...	1.064346...	1.653296...	1.376239...	0.215477...	0.349749...
Seed_280	subTeV	26.32500...	-0.41999...	0.100000...	1.0	0.0			Cavity	0.552785...					1.0	0.146558...	221.3659...		1.431983...	2.407378...		0.002449...	
Seed_298	postTeV	20.53499...	-1.95000...	0.119999...	0.71875	3.0	Seed_515	Seed_515	Peak	-0.98797...					1.0	0.131800...	13.35802...	-16.0443...	1.078465...	2.351605...	2.348702...	0.425466...	0.149609...
Seed_342	postTeV	20.80500...	-2.43000...	0.150000...	0.875	3.0	Seed_515	Seed_515	Peak	-0.95454...					1.0	0.064308...	0.888447...	-28.5139...	9.906006...	2.713505...	2.348702...	0.587778...	0.149609...
Seed_346	postTeV	21.45000...	-1.48500...	0.150000...	0.75	4.0	Seed_515	Seed_440	Peak	-0.97192...					1.0	0.067309...	8.081151...	-21.3212...	9.705660...	2.378850...	2.348702...	0.164126...	0.149609...
Seed_348	postTeV	24.16500...	-1.35000...	0.150000...	1.0	0.0			Peak	-0.97574...	composit...	0.069892...	0.528571...	25.63436...	1.0	0.264757...	0.982327...		5.959616...	1.762806...		0.999931...	
Seed_351	postTeV	24.97500...	-1.08000...	0.150000...	1.0	0.0			Peak	-0.99118...	pwn_262	0.034901...	0.663934...	54.79167...	1.0	0.287974...	0.524980...		6.023434...	1.337375...		0.601013...	
Seed_373	postTeV	19.97999...	-0.08250...	0.180000...	0.84375	3.0	Seed_492	Seed_492	Peak	-0.97487...	pwn_62	0.083953...	0.468253...	9.337107...	1.0	0.118893...	0.599142...	-0.24573...	4.492718...	1.064530...	1.376239...	0.635469...	0.349749...
Seed_387	postTeV	19.17000...	0.194999...	0.150000...	1.0	0.0			Peak	-0.98379...	pwn_464	0.051478...	0.576470...	7.675066...	1.0	0.285271...	0.259057...		1.855192...	1.365402...		0.197836...	
Seed_437	postTeV	21.27000...	-1.91999...	0.330000...	0.535714...	4.0	Seed_515	Seed_515	Cavity	0.888015...					-1.0	-0.02767...	0.044890...	-29.3575...	4.505994...	1.330313...	2.348702...	0.272173...	0.149609...
Seed_440	postTeV	21.40500...	-1.59000...	0.270000...	0.517857...	3.0	Seed_515	Seed_515	Peak	-0.35065...					-1.0	-0.02009...	19.85866...	-9.54372...	2.123508...	2.427763...	2.348702...	0.108770...	0.149609...
Seed_477	postTeV	26.81999...	-0.25499...	0.360000...	0.75	1.0			Peak	-0.97313...	HESS J18...	0.062554...	0.333581...	18273.8125	0.0	0.125852...	9938.805...		3.935406...	2.384824...		0.011623...	
Seed_492	postTeV	20.17499...	0.0	0.330000...	0.5	1.0			Cavity	0.357750...	snr_227	0.109625...	0.593548...	-0.00325...	-1.0	-0.01716...	0.844881...		1.693965...	1.376239...		0.349749...	

Spectral extraction for each seed or full maps

- MapDataset.to_spectrum_dataset slow because of https://docs.gammapy.org/dev/_modules/gammapy/maps/wcsnd.html#WcsNDMap.get_spectrum

- Instead convolution with kernel of seed size to extract counts and background,
  IndexMapEstimator: Similar approach can be used to produce spectral index maps,
  Could be useful for structures separation based on different spectral properties



Separation of few known source using hierarchical clustering on spectral index maps computed for Fermi-LAT maps

Models management

- Most of the modeling and fitting steps are model management and parameter freezing/unfreezing

Could propose some macro to perfom these faster, not built-in in gammapy, but as user-contributed scripts ?
(so we don't have to maintain that and encourage user exchanges)

For example merging sources outside maskfit as one background

```
# build local model, caching outside sources as bkgs
for m in tmp_models:
    m.datasets_names = []
for dataset in datasets:
    # bkg sources
    geom2d = dataset.mask_fit.sum_over_axes().geom
    npred_out = Map.from_geom(dataset.counts.geom)
    for model in sources_models:
        pos = model.position.galactic
        iy, ix = geom2d.coord_to_pix((pos.l, pos.b))
        ix = int(np.rint(ix))
        iy = int(np.rint(iy))
        if not np.any(dataset.mask_fit.data[:, ix, iy]):
            # if center not in mask_fit added to background sources
            npred_out.stack(dataset._evaluators[model]._npred_cached)
        else:
            # otherwise included in this dataset
            model.datasets_names.append(dataset.name)
    if npred_out.data.sum() != 0:
        name = "outer_sources_" + dataset.name
        filename = roidir + name + ".fits"
        npred_out.write(filename, overwrite=True)
        bkg = BackgroundModel(
            npred_out,
            name=name,
            datasets_names=[dataset.name],
            filename=filename,
        )
        if freeze_bkg_sources:
            bkg.parameters.freeze_all()
        else:
            bkg.tilt.frozen = True
        bkgs.append(bkg)

for m in tmp_models:
    if m.datasets_names == []:
        sources_models.remove(m)  # remove source that are bkg in all datasets
```

Fitting by group of over-lapping sources

```
for name, mgrp in mgroups.items():
    print(f"Nest: {name}: {len(mgrp)} objects")
    frozen_params = []
    for m in datasets.models:
        if m in mgrp:
            #the source in the group are not frozen
            if m.name==name:
                #also release more parameters of the main object
                m.spatial_model.eta.frozen = False
                m.spatial_model.e.frozen = False
                m.spatial_model.phi.frozen = False
        else:
            for p in m.parameters:
                if p.frozen == False:
                    p.frozen = True
                    frozen_params.append(p)
    nfree = sum([not p.frozen for p in datasets.models.parameters])

    if optimize_opts["backend"] == "minuit":
        results = Fit(datasets).run(optimize_opts=optimize_opts)
    else:
        optimize_opts["maxfev"] = fevfree * nfree
        results = Fit(datasets).optimize(**optimize_opts)
    print(results)

    # unfreeze parameters to retore previous status for next group
    for p in frozen_params:
        p.frozen = False
```

Summary

- Missing models for ctools compatibility

- No real limitations:

Most of the extra functionalities needed can be built on top of gammapy
or using other python module conjointly with gammapy (as scikit-image and scikit-learn)

- Spectral extraction and flux points require re-work to improve performance

- lru_cache() in map.geom break usage of multiprocessing
  for example in mcmc_sampling cannot use more than one core

- How to encourage user contributions ?