Towards the homogenization of parameters within the estimators module

Bruno Khelifi

July 9, 2020

Contents

This note explains how the 'sigma' parameters are used in different estimators. This work aims to unify their usage and synthaxe.

1 What we have

- ExcessMapEstimator
  - n_sigma (float): Confidence level for the asymmetric errors expressed in number of sigma.
  - n_sigma_ul (float): Confidence level for the upper limits expressed in number of sigma.
  - Note: the output object in not an astropy.Table -> not consistant, no metadata of the confidence level for ULs, no 'sigma' threshold for the UL computation (up to the user to choose it from the output)
- FluxPointsEstimator
  - sigma (int): Sigma to use for asymmetric error computation.
  - sigma_ul (int): Sigma to use for upper limit computation.

  - Note: call of 'FluxEstimator' (see below), no 'sigma' threshold for the UL computation (up to the user to choose it from the output)

- FluxEstimator
  - sigma (int): Sigma to use for asymmetric error computation.
  - sigma_ul (int): Sigma to use for upper limit computation.
  - Note: here sigma_ul is not well described (the confidence level or the sigma threshold?), call of 'ParameterEstimator' (see below)

- LightCurveEstimator
  - sigma (int): Sigma to use for asymmetric error computation.
  - sigma_ul (int): Sigma to use for upper limit computation.
  - Note: usage of 'FluxEstimator'

- ParameterEstimator
  - sigma (int): Sigma to use for asymmetric error computation.
  - sigma_ul (int): Sigma to use for upper limit computation.
  - 'err': from the fit result directly, not using sigma!; whereas 'errp-errn' uses sigma via self.fit.confidence -> not consistant
  - 'ul': uses sigma_ul from self.fit.confidence -> not using stats.compute_upper_limit(self.n_si
  - 'ts': Likelihood(null value) - Likelihood(minimum)
  - Note: same comments than for 'FluxEstimator'

- ASmoothMapEstimator
  - threshold (float): Significance threshold
  - Note: the so call significance is neither a cash or wstat one! (counts - background) / np.sqrt(counts + background) -> not consistant, the threshold value is not stored as meta-data in the final map

- TSMapEstimator
  - error_sigma (int): Sigma for flux error
  - ul_sigma (int): Sigma for flux upper limits.

  - threshold (float): If the TS value corresponding to the initial flux estimate is not above this threshold, the optimizing step is omitted to save computing time. -> not consistant with 'ASmoothMapEstimator': Threshold on (Cash_0 - Cash_1), not on significance
  - Note: UL as result["flux"] + ul_sigma * result["flux_err"] -> Not consistant with the 'ParameterEstimator' method and not using stats.compute_upper_limit(self.n_sigma_ul), threshold not stored as metadata

- SensitivityEstimator
  - sigma (float, optional): Minimum significance
- ExcessProfileEstimator
  - n_sigma (float, optional): Number of sigma to compute errors.
  - n_sigma_ul (float, optional): Number of sigma to compute upper limit.
  - Note: flux_err and flux_ul algorithms different from the TSMapEstimator, not storing the confidence level in metadata, no 'sigma' threshold for the UL computation (up to the user to choose it from the output)
- modeling.fit
  - sigma (float): Number of standard deviations for the confidence level
  - Note: significance comptuted with stats.excess_matching_significance
- stats.counts_statistics
  - in excess_matching_significance, significance (float): Significance
  - elsewhere usage of n_sigma (float)

2 Analysis

- The names for the same paramater types can be different (e.g. error_sigma, n_sigma, etc)

- We can have the same parameter name for different parameter types (e.g. threshold)
- We have different default values according to the estimators (e.g. n_sigma_ul 2 or 3)
- All estimators except 'ExcessMapEstimator' use an astropy.Table
- The threshols are not always stored as meta-data
- Not using the same algo to compute ULs
- It seems that we are not always using the same algorithms for the error and ULs computations (TBC)

3 Proposals

- For all the above-mentionned classes (even in modeling):
  - n_sigma (float, default: 1): Number of sigma to compute errors
  - n_sigma_ul (float, default: 2): Number of sigma (confidence level) to compute upper limit OR conf_level (float, default: 0.9545): confidence level to compute ULs
  - n_sigma_th_ul (float, default: 3): Number of sigma (confidence level) above which to compute upper limit
  - ExcessMapEstimator: change the output from dict to astropy.Table
  - ParameterEsimator: usage of stats.compute_upper_limit?
  - TSMapEstimator and ASmoothMapEstimator: usage of our stat class
  - add in tables the metadata (n_sigma, n_sigma_ul, n_sigma_th_ul)