





HYPER-PY: HYbrid Photometry and Extraction Routine in PYthon

Alessio Traficante¹, Fabrizio De Angelis¹, Alice Nucara¹, and Milena Benedettini¹

¹ INAF-IAPS, Via Fosso del Cavaliere, 100, 00133 Rome (IT)  Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: 

Submitted: 03 September 2025

Published: unpublished

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/))

Summary

Source extraction and photometry of compact objects are fundamental tasks in observational astronomy. Over the years, various tools have been developed to address the inherent complexity of astronomical data—particularly for accurate background estimation and removal, and for deblending nearby sources to ensure reliable flux measurements across multiple wavelengths (e.g. Cutex: Molinari et al. (2011); getsources: Men'shchikov et al. (2012); Fellwalker: Berry (2015); *Astrodendro*). These challenges are especially pronounced in star-forming regions, which are best observed in the far-infrared (FIR), sub-millimeter, and millimeter regimes, where the cold, dense envelopes of compact sources emit most strongly. To address these needs, several software packages have been designed to handle the structured backgrounds and blended source populations typical of observations with instruments such as Herschel (70–500 μm) and ALMA (1–3 mm). These packages differ significantly in their detection strategies and flux estimation methods. In this framework, we developed **HYPER** (HYbrid Photometry and Extraction Routine, Traficante et al. (2015)), originally implemented in Interactive Data Language (IDL), with the goal of providing robust and reproducible photometry of compact sources in FIR/sub-mm/mm maps. **HYPER** combines: (1) source detection via high-pass filtering; (2) background estimation and removal through local polynomial fitting; and (3) source modeling using 2D elliptical Gaussians. For blended regions, **HYPER** fits multiple Gaussians simultaneously to deblend overlapping sources, subtracting companions before performing photometry. Aperture photometry in **HYPER** is then carried out on the background-subtracted, companion-subtracted images, using the footprint defined by each source's 2D Gaussian model. This ensures a consistent and robust integrated flux measurement, even in crowded or strongly structured environments Traficante et al. (2015); Traficante et al. (2023). The hybrid nature of **HYPER** lies in its combined approach: using 2D Gaussian modeling, while retaining classical aperture photometry techniques. In this work, we present **Hyper-Py**, a fully restructured and extended version of **HYPER** developed entirely in Python. *Hyper-Py* not only replicates the core logic of the original IDL implementation, but introduces multiple improvements in performance, configurability, and background modeling capabilities—making it a modern and flexible tool for source extraction and photometric analysis across a wide range of datasets. Notably, *Hyper-Py* also introduces the ability to estimate and subtract the background emission across individual slices of 3D datacubes, enabling consistent background modeling along the spectral axis for line or continuum studies in spectrally resolved observations.

Statement of need

Hyper-Py is a Python package freely accessible to the community. This new Python implementation is a conversion of the IDL version **HYPER** which includes several improvements to the original package:

42 **Parallel execution for multi-map analysis.** *Hyper-Py* introduces built-in parallelization for the
43 analysis of multiple input maps. On multi-core systems, each map is independently assigned
44 to a dedicated core, allowing concurrent execution of the full photometric pipeline—source
45 detection, background fitting, Gaussian modeling, and aperture photometry—for each map.
46 This parallel framework substantially improves computational efficiency without altering the
47 scientific output for individual maps.

48 **Native support for the analysis of FITS datacubes.** The code enables users to treat each slice
49 along the third axis as an independent 2D map. This functionality is fully compatible with
50 the existing parallelization framework, allowing multiple slices to be processed simultaneously
51 across different cores. The primary goal of this mode is to estimate the spatially-varying
52 background emission on a per-slice basis. The final output is a reconstructed 3D background
53 datacube with the same shape as the input cube. This background cube can either be focused
54 on a specific region or line of sight—leaving all other voxels as NaNs—or computed across
55 the full spatial extent of the cube. The region where to perform the extraction and related
56 parameters are configurable via the `config.yaml` file, offering flexibility for both targeted and
57 global background modeling.

58 **Improved source detection reliability.** The source detection module has been significantly
59 improved through the implementation of a more robust sigma-clipping algorithm for estimating
60 the root mean square (*rms*) noise of the input map. This enhancement ensures a more
61 accurate characterization of the background fluctuations by iteratively excluding outliers and
62 recalculating the noise level, even in the presence of bright sources or residual structures. The
63 resulting *rms* value is then used as a reference threshold to identify compact sources with peak
64 intensities exceeding a user-defined significance level, set as $n_sigma \times rms$, where *n_sigma* is
65 configurable via the `config.yaml` file. This refinement improves the reliability and reproducibility
66 of the source detection process across heterogeneous datasets.

67 **Advanced background estimation strategy.** *Hyper-Py* introduces a robust and flexible back-
68 ground estimation framework designed to improve subtraction accuracy in complex regions and
69 in case of blended sources. Unlike the original IDL version, which estimated and subtracted
70 the background independently of source modeling Traficante et al. (2015), *Hyper-Py* supports
71 multiple statistical methods for background fitting, applied to masked cutouts around each
72 source. Users can select from least-squares regression, Huber regression, and Theil–Sen re-
73 gression, either individually or in combination. The least-squares method is optimal in regions
74 dominated by Gaussian noise. The Huber regressor provides robustness against outliers by
75 interpolating between L2 and L1 loss functions, with the tuning parameter ϵ (`huber_`
76 `epsilon` in the `config` file) controlling the transition. The Theil–Sen estimator is a non-parametric,
77 highly robust approach particularly suited for non-Gaussian noise or residual contamination.
78 When multiple methods are enabled, *Hyper-Py* evaluates all and selects the background model
79 that minimizes the residuals within the unmasked region, ensuring accurate reconstruction
80 even in the presence of variable gradients or faint extended emission. In addition, *Hyper-Py*
81 offers an optional joint fit of the background and 2D elliptical Gaussians, which may improve
82 stability or convergence in specific cases. When this combined fit is used, the background
83 polynomial terms can be regularized using L2 (ridge) regression, helping suppress unphysically
84 large coefficients. This constraint enhances the robustness of the background model in regions
85 with strong intensity gradients or spatially variable emission, reducing the risk of overfitting
86 the background at the expenses of the source flux estimation.

87 **Gaussian + background model optimization strategy.** The combined fitting is optimized using
88 the Levenberg–Marquardt algorithm, implemented via the “`least_squares`” minimizer from the
89 `lmfit` package (equivalent to the `scipy.optimize.least_squares` function). This method allows
90 flexible control over the cost function via the `loss` parameter, which modifies the residuals used
91 during minimization. Specifically, we adopt a robust loss function, setting `loss = “cauchy”`. This
92 choice reduces the influence of outliers by scaling the residuals in a non-linear way, effectively
93 down-weighting pixels that deviate significantly from the model. Compared to the standard
94 least-squares loss (“linear”), robust losses like “cauchy” are better suited to data affected

by small-scale artifacts, unmasked sources, or non-Gaussian noise features. Alternative loss functions such as “soft_l1” and “huber” are also available and may offer improved convergence speed in some cases, particularly when dealing with moderately noisy data.

Model selection criteria for background fitting. In *Hyper-Py*, the optimal background model—i.e., the best combination of box size and polynomial order—is determined by evaluating the residuals between the observed data and the fitted model. The framework offers a configurable choice of model selection criteria, specified via the `config.yaml` file. Users can select from three statistical metrics: Normalized Mean Squared Error (NMSE), reduced chi-squared (χ^2_ν), or the Bayesian Information Criterion (BIC). By default, *Hyper-Py* adopts NMSE, a robust, unitless, and scale-independent metric that quantifies the fraction of residual power relative to the total signal power in the cutout. Unlike reduced chi-squared, NMSE is not sensitive to changes in pixel weighting schemes or the number of valid (unmasked) pixels, making it particularly reliable when background fits are performed under varying masking conditions, inverse-rms weighting, or SNR-based weighting. In contrast, the reduced chi-squared statistic depends directly on the assumed noise model and weighting, and may therefore be biased when the number or distribution of contributing pixels changes. The BIC criterion offers an alternative that penalizes overfitting by incorporating the number of model parameters and the sample size, favoring simpler models when multiple fits achieve similar residuals. This flexibility allows users to tailor the model selection strategy to the scientific context or noise characteristics of their data, ensuring that the chosen background model is both statistically sound and physically meaningful.

Improved user configurability. *Hyper-Py* is designed to be more user-friendly, featuring a clear and well-documented configuration file. This allows users to adapt the full photometric workflow to a wide range of observational conditions and scientific goals by modifying only a minimal set of parameters. The modular structure of the configuration also enhances transparency and reproducibility in all stages of the analysis.

We assessed the performance of the *Hyper-Py* pipeline using a dedicated suite of simulations. Specifically, we adopted a noise-only map derived from the ALMA program #2022.1.0917.S and produced two maps with this reference header. In each of these maps we injected a variable background and 500 synthetic 2D Gaussian sources into it. These sources were designed to emulate the properties of real, compact astronomical objects: integrated fluxes ranged between 8 and 20 times the map *rms* (corresponding to peak fluxes of approximately 1–1.5 times the *rms*), and FWHMs spanned from 0.5 to 1.5 times the beam size, as computed from the FITS header, to simulate both unresolved and moderately extended sources Elia et al. (2021). The source position angles were randomly assigned, and a minimum overlap fraction of 30% was imposed to ensure a significant level of blending, thus providing a rigorous test of the code under realistic and challenging conditions. We then compared the performance of the original IDL implementation of **HYPER** with the new Python-based *Hyper-Py* version. Both codes were run using equivalent configurations, with *Hyper-Py* additionally benefiting from its extended capabilities—such as improved background estimation, optional L2 regularization, and multi-core parallel processing. The main results of this comparison are presented in **Table 1**, where we show the differences between the number of identified sources and the false positives by *Hyper-Py* and **HYPER**, respectively.

Catalog	Source Type	Total	Matched	False	False Percentage
1	<i>Hyper-py</i>	500	490	4	0.8%
1	HYPER (IDL)	500	493	73	12.9%
2	<i>Hyper-py</i>	500	487	4	0.8%
2	HYPER (IDL)	500	487	46	8.6%

In addition, Figure 1 and Figure 2 show the differences between the peak fluxes and the integrated fluxes of the sources with respect to the reference values of the simulated input

140 sources as estimated by *Hyper-Py* and HYPER, respectively.

141 *Hyper-Py* is freely available for download via its [GitHub repository](#), and can also be installed
142 using pip, as described in the accompanying README file.

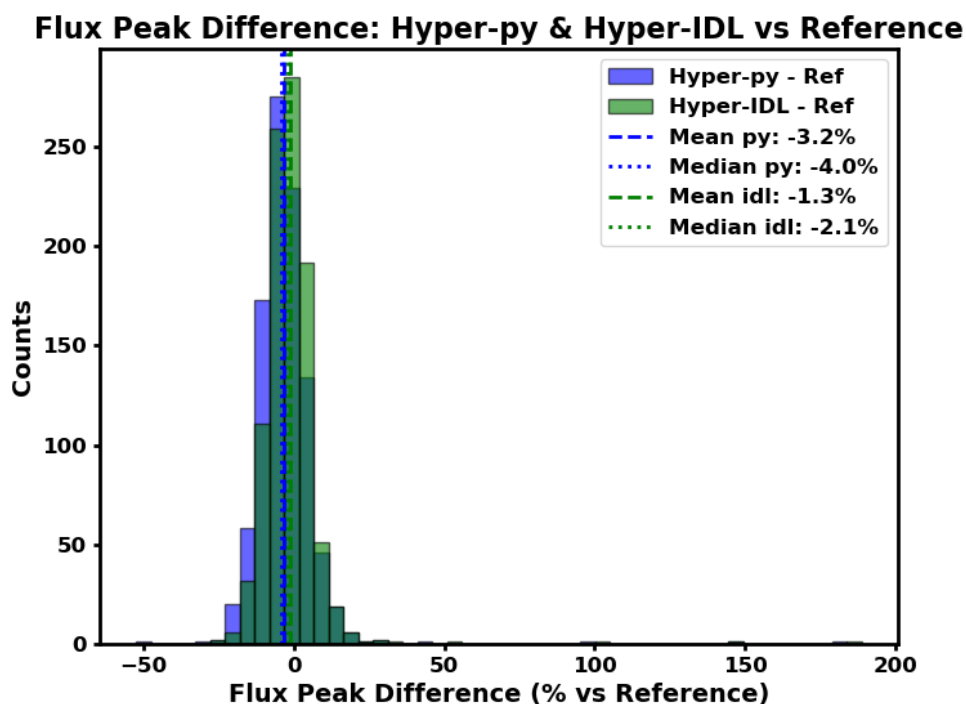


Figure 1: histogram of the differences between the peak fluxes of the sources as recovered by *Hyper-Py* and HYPER, respectively, with respect to the reference values of the simulated input sources

Integrated Flux Difference: Hyper-py & Hyper-IDL vs Reference

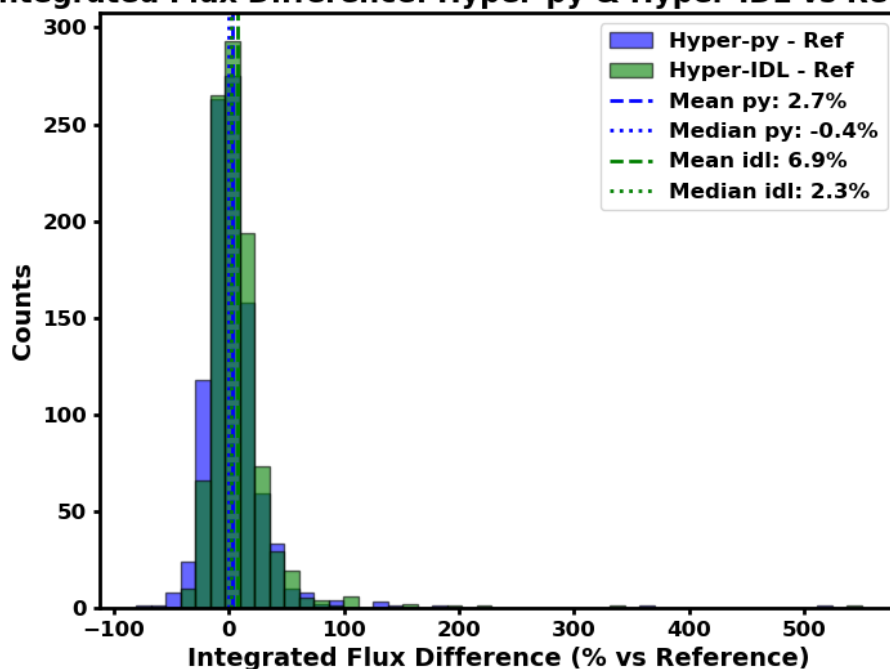


Figure 2: histogram of the differences between the peak fluxes of the sources as recovered by *Hyper-Py* and *HYPER*, respectively, with respect to the reference values of the simulated input sources

- 143 Berry, D. S. (2015). FellWalker-A clump identification algorithm. *Astronomy and Computing*,
144 10, 22–31. <https://doi.org/10.1016/j.ascom.2014.11.004>
- 145 Elia, D., Merello, M., Molinari, S., Schisano, E., Zavagno, A., Russeil, D., Mège, P., Martin, P.
146 G., Olmi, L., Pestalozzi, M., Plume, R., Ragan, S. E., Benedettini, M., Eden, D. J., Moore,
147 T. J. T., Noriega-Crespo, A., Paladini, R., Palmeirim, P., Pezzuto, S., ... Polychroni, D.
148 (2021). The Hi-GAL compact source catalogue - II. The 360° catalogue of clump physical
149 properties. *MNRAS*, 504(2), 2742–2766. <https://doi.org/10.1093/mnras/stab1038>
- 150 Men'shchikov, A., André, Ph., Didelon, P., Motte, F., Hennemann, M., & Schneider, N.
151 (2012). A multi-scale, multi-wavelength source extraction method: getsources. *Astronomy
152 & Astrophysics*, 542, A81. <https://doi.org/10.1051/0004-6361/201218797>
- 153 Molinari, S., Schisano, E., Faustini, F., Pestalozzi, M., di Giorgio, A. M., & Liu, S. (2011).
154 Source extraction and photometry for the far-infrared and sub-millimeter continuum in the
155 presence of complex backgrounds. *Astronomy & Astrophysics*, 530, A133.
- 156 Traficante, A., Fuller, G. A., Peretto, N., & Pineda, J. E. (2015). High-resolution maps of
157 star-forming regions - i. The HYPER algorithm for compact source extraction. *Monthly
158 Notices of the Royal Astronomical Society*, 451(4), 4086–4103. <https://doi.org/10.1093/mnras/stv1221>
- 160 Traficante, A., Jones, B. M., Avison, A., Fuller, G. A., Benedettini, M., Elia, D., Molinari, S.,
161 Peretto, N., Pezzuto, S., Pillai, T., Rygl, K. L. J., Schisano, E., & Smith, R. J. (2023).
162 The SQUALO project (Star formation in QUIescent And Luminous Objects) I: clump-fed
163 accretion mechanism in high-mass star-forming objects. *MNRAS*, 520(2), 2306–2327.
164 <https://doi.org/10.1093/mnras/stad272>