The Journal of Open Source Software

PyAutoCTI: Open-Source Charge Transfer Inefficiency Calibration

James. W. Nightingale [1], **Richard J. Massey** [1], **Jacob Kegerreis** [2], **and Richard G. Hayes**[1]

1 Institute for Computational Cosmology, Stockton Rd, Durham DH1 3LE **2** NASA Ames Research Center, Moffett Field, CA 94035, USA

DOI: 10.21105/joss.04904

Software

- Review
- Repository
- Archive

Editor: Axel Donath

Reviewers:

- @jryon
- @mwcraig

Submitted: 02 August 2022
Published: 01 June 2024

Summary

Over the past half a century, space telescopes have given us an extraordinary view of the Universe, for example detecting the faint spectra of the first galaxies in the Universe (Bouwens et al., 2015; Dunlop et al., 2013), measuring tiny distortions in galaxy shapes due to gravitational lensing (Massey et al., 2007; Schrabback et al., 2010) and a high precision map of stars within the Milky Way (Brown et al., 2018; Gaia Collaboration et al., 2021). These observations require a deep understanding of the telescope's instrumental characteristics, including the calibration and correction of charge transfer inefficiency (hereafter CTI, Massey et al., 2010, 2014), a phenomenon where radiation damage to the telescope's charge-coupled device (CCD) sensors leads to gradually increased smearing in acquired exposures over the telescope's lifetime.

`PyAutoCTI` is an open-source Python 3.8 - 3.11 (Van Rossum & Drake, 2009) package for the calibration of CTI for space telescopes. By interfacing with `arCTIc` (`the algorithm for Charge Transfer Inefficiency correction`) the calibrated CTI models can straightforwardly be used to correct and remove CTI in every science image taken throughout the telescope's lifetime. Core features include fully automated Bayesian model-fitting of CTI calibration data, support for different calibration datasets such as warm pixels used for Hubble Space Telescope calibration (Massey et al., 2010; Massey, 2010) and a database for building a temporal model of CTI over the telescope's lifetime. The software places a focus on big data analysis, including support for graphical models that simultaneously fits large CTI calibration datasets and an SQLite3 database that allows extensive suites of calibration results to be loaded, queried and analysed. Accompanying `PyAutoCTI` is the autocti workspace, which includes example scripts, datasets and an overview of core `PyAutoCTI` functionality. Readers can check out the readthedocs for a complete overview of `PyAutoCTI`'s features.

Background

During an exposure, photons hitting a CCD generate photo-electrons that are held in discrete clouds (pixels) by a grid of electrostatic potentials. At the end of the exposure, the potentials are varied so as to move clouds of electrons first in the 'parallel' direction to the edge of the CCD, then along the perpendicular 'serial' register to an amplifier and read-out electronics that count the number of electrons. The CCD substrate through which the electrons move is built of a silicon lattice. However, defects in the lattice known as 'traps' can temporarily capture electrons. If this time is longer than the time taken to move a cloud of electrons to the next pixel, an electron may not be released back into its original charge cloud, but to a subsequent one: creating a characteristic 'trailing' or 'smearing' effect behind sources in the image. The CTI properties of a CCD can be calibrated, using image data that has sharp edges whose position and flux are known (or can be determined) prior to readout. `PyAutoCTI` fits a

CTI model to these sharp edges and their preceding trails which form due to CTI.

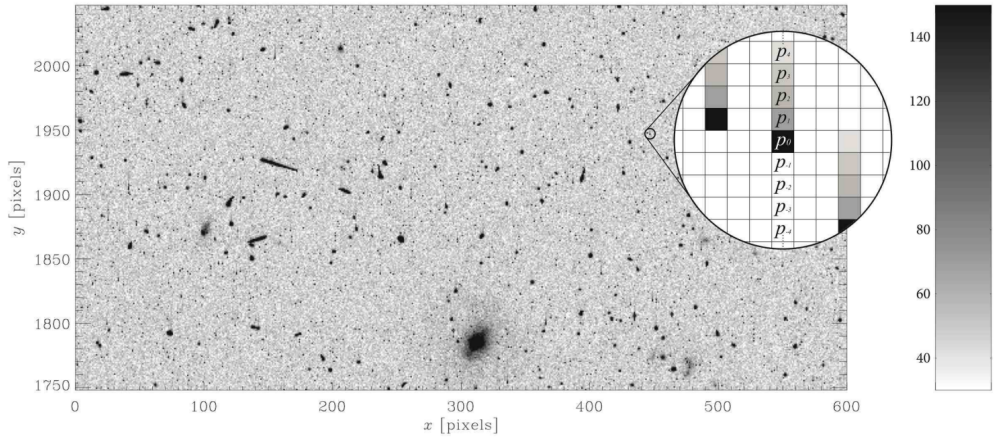

Figure 1: A typical, raw HST ACS / WFC image, in units of electrons. This 30" x 15" (600 x 300 pixels) region contains warm pixels, with an example warm pixel towards the right zoomed in on. This reveals CTI trailing behind (above) the warm pixel, which `PyAutoCTI` fits to calibrate CTI. This figure is taken from (Massey et al., 2010) and has been used with the author's permission.

Figure 1 shows an example of the CTI calibration data which `PyAutoCTI` was originally developed to fit – warm pixels in the Hubble Space Telescope (Massey et al., 2010). Warm pixels are similar to short circuits in the CCD electrostatic potentials used to collect charge, which continuously inject spurious charge into specific pixels. These should appear as delta functions in the data, with CTI trails appearing in Figure 1 in the parallel clocking direction (upwards in Figure 1) next to each warm pixel. `PyAutoCTI` has dedicated functionality for fitting 1D calibration datasets such as those extracted from warm pixels. `PyAutoCTI` also supports for 2D calibration datasets, for example charge injection line imaging calibration data, which will be used to calibrate CTI for the European Space Agency's Euclid space mission. After CTI calibration has inferred an accurate CTI model, this can be passed to `arCTIc` in order to correct CTI from science imaging data.

Statement of Need

Space based telescopes planned for launch over the coming decades will map out the Universe's dark matter via weak gravitational lensing (Massey et al., 2007), make precision detections of exoplanets that are further from Earth than ever seen before (Halverson et al., 2016) and perform astrometry of moving objects (Bellini et al., 2013). For these ambitious projects to be successful they all require CTI is removed from science imaging with very stringent requirements, necessitating that CTI calibration provides in-depth knowledge about CTI on every CCD. `PyAutoCTI` ensures that large CTI calibration datasets can be exploited to measure the CTI model over the telescope's life and it provides tools which maximally extract information from these datasets using contemporary Bayesian inference techniques. The intended audience for `PyAutoCTI` is therefore the astronomical community, including instrument scientists, who are responsible for maintaining the calibration of space telescopes and the scientific community, who can use `PyAutoCTI` to perform CTI calculations for their science data.

The majority of software packages for CTI, such as `arCTIc` (Massey et al., 2014), `C3TM` (Skottfelt et al., 2018) and `CDM03` (Short et al., 2010), do not perform CTI calibration. They instead model the CCD clocking procedure, including a CTI model describing the traps on a CCD, enabling the addition and correction of CTI to imaging data. `PyAutoCTI` wraps one of these algorithm (specifically `arCTIc`) in order to measure the CTI model from calibration data.

Hubble Space Telescope CTI calibration uses an algorithm developed by STScI (Anderson et al., 2021; Anderson & Bedin, 2010; Anderson & Ryon, 2018). The open-source software `Pyxel` (https://esa.gitlab.io/pyxel/page/introduction/) has a "calibration mode" which also wraps these algorithms to perform CTI calibration. However, `Pyxel` is more general purpose and has tools for the entire CCD clocking procedure (e.g. bias, dark frames, flat fields). `PyAutoCTI` is therefore more specialized and has a more extensive suite of tools for CTI calibration.

Software API and Features

At the heart of the `PyAutoCTI` API are `Trap` objects, which represent the populations of traps on a CCD which cause CTI. The volume filling behaviour of a CCD is modeled via `CCD` objects, which are combined with traps to compose CTI models which add CTI to a mock CTI calibration data via `arCTIc`. `PyAutoCTI` has dedicated objects for specific CTI calibration datasets and bespoke tools for visualizing these datasets and masking them before fitting.
The `astropy` (Astropy Collaboration et al., 2013; Price-Whelan et al., 2018) cosmology module handles unit conversions and calculations are optimized using the packages `NumPy` (van der Walt et al., 2011) and `numba` (Lam et al., 2015). Dependencies also include `scikit-image` (Van der Walt et al., 2014), `scikit-learn` (Pedregosa et al., 2011) and `Scipy` (Virtanen et al., 2020).

To perform model-fitting, `PyAutoCTI` adopts the probabilistic programming language `PyAutoFit`[1] (Nightingale et al., 2021). `PyAutoFit` allows users to compose a CTI model from `Trap` and `CCD` objects, customize the model parameterization and fit it to data via a non-linear search, for example `dynesty` (Speagle, 2020), `emcee` (Foreman-Mackey et al., 2013) or `PySwarms` (Miranda, 2018), with visualization performed by `Matplotlib` (Hunter, 2007) and `corner.py` (Foreman-Mackey, 2016). `PyAutoFit`'s graphical modeling framework allows one to fit a temporal model to a suite of CTI calibration data. Using a technique called expectation propagation (Vehtari et al., 2020), the framework fits each dataset one-by-one and combines the results of every fit into a temporal model using a self-consistent Bayesian framework. To ensure the analysis and interpretation of fits to large datasets is feasible, `PyAutoFit`'s database tools write modeling results to a relational database which can be queried from a storage drive to a Python script or Jupyter notebook. This uses memory-light `Python` generators, ensuring it is practical for use over a telescope's entire lifetime, where CTI calibration data taken with a daily cadence may consist of thousands of datasets.

Workspace

`PyAutoCTI` is distributed with the autocti workspace, which contains example scripts for modeling and simulating CTI. The workspace is accessible on Binder and example scripts can therefore be run without a local `PyAutoCTI` installation.

Acknowledgements

JWN and RJM are supported by the UK Space Agency, through grant ST/V001582/1, and by InnovateUK through grant TS/V002856/1. RGH is supported by STFC Opportunities grant ST/T002565/1. RJM is supported by a Royal Society University Research Fellowship. This work used the DiRAC@Durham facility managed by the Institute for Computational Cosmology on behalf of the STFC DiRAC HPC Facility (www.dirac.ac.uk). The equipment was funded by BEIS capital funding via STFC capital grants ST/K00042X/1, ST/P002293/1, ST/R002371/1 and ST/S002502/1, Durham University and STFC operations grant ST/R000832/1. DiRAC is part of the National e-Infrastructure.

[1] https://github.com/rhayes777/PyAutoFit

References

Anderson, J., Baggett, S., & Kuhn, B. (2021). *Updating the WFC3/UVIS CTE model and Mitigation Strategies* (p. 9). Instrument Science Report 2021-9, 44 pages.

Anderson, J., & Bedin, L. R. (2010). An Empirical Pixel-Based Correction for Imperfect CTE. I. HST's Advanced Camera for Surveys. *PASP*, *122*(895), 1035. https://doi.org/10.1086/656399

Anderson, J., & Ryon, J. E. (2018). *Improving the Pixel-Based CTE-correction Model for ACS/WFC* (p. 4). Instrument Science Report ACS 2018-04, 37 pages.

Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., Greenfield, P., Droettboom, M., Bray, E., Aldcroft, T., Davis, M., Ginsburg, A., Price-Whelan, A. M., Kerzendorf, W. E., Conley, A., Crighton, N., Barbary, K., Muna, D., Ferguson, H., Grollier, F., Parikh, M. M., Nair, P. H., … Streicher, O. (2013). Astropy: A community Python package for astronomy. *A&A*, *558*, A33. https://doi.org/10.1051/0004-6361/201322068

Bellini, A., van der Marel, R. P., & Anderson, J. (2013). HST proper motions of stars within globular clusters. *Mem. Societa Astronomica Italiana*, *84*, 140. https://doi.org/10.48550/arXiv.1301.2338

Bouwens, R. J., Illingworth, G. D., Oesch, P. A., Trenti, M., Labbé, I., Bradley, L., Carollo, M., Van Dokkum, P. G., Gonzalez, V., Holwerda, B., Franx, M., Spitler, L., Smit, R., & Magee, D. (2015). UV luminosity functions at redshifts z 4 to z 10: 10,000 galaxies from HST legacy fields. *ApJ*, *803*(1), 1–49. https://doi.org/10.1088/0004-637X/803/1/34

Brown, A. G. A., Vallenari, A., Prusti, T., De Bruijne, J. H. J., Babusiaux, C., Bailer-Jones, C. A. L., Biermann, M., Evans, D. W., Eyer, L., Jansen, F., Jordi, C., Klioner, S. A., Lammers, U., Lindegren, L., Luri, X., Mignard, F., Panem, C., Pourbaix, D., Randich, S., … Zwitter, T. (2018). Summary of the contents and survey properties. *A&A*, *616*. https://doi.org/10.1051/0004-6361/201833051

Dunlop, J. S., Rogers, A. B., McLure, R. J., Ellis, R. S., Robertson, B. E., Koekemoer, A., Dayal, P., Curtis-Lake, E., Wild, V., Charlot, S., Bowler, R. A. A., Schenker, M. A., Ouchi, M., Ono, Y., Cirasuolo, M., Furlanetto, S. R., Stark, D. P., Targett, T. A., & Schneider, E. (2013). The UV continua and inferred stellar populations of galaxies at z 7-9 revealed by the Hubble Ultra-Deep Field 2012 campaign. *MNRAS*, *432*(4), 3520–3533. https://doi.org/10.1093/mnras/stt702

Foreman-Mackey, D. (2016). Corner.py: Scatterplot matrices in python. *The Journal of Open Source Software*, *1*(2), 24. https://doi.org/10.21105/joss.00024

Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. (2013). emcee : The MCMC Hammer. *Publications of the Astronomical Society of the Pacific*, *125*(925), 306–312. https://doi.org/10.1086/670067

Gaia Collaboration, Brown, A. G. A., Vallenari, A., Prusti, T., de Bruijne, J. H. J., Babusiaux, C., Biermann, M., Creevey, O. L., Evans, D. W., Eyer, L., Hutton, A., Jansen, F., Jordi, C., Klioner, S. A., Lammers, U., Lindegren, L., Luri, X., Mignard, F., Panem, C., … Zwitter, T. (2021). Gaia early data release 3 - summary of the contents and survey properties (corrigendum). *A&A*, *650*, C3. https://doi.org/10.1051/0004-6361/202039657e

Halverson, S., Terrien, R., Mahadevan, S., Roy, A., Bender, C., Stefánsson, G. K., Monson, A., Levi, E., Hearty, F., Blake, C., McElwain, M., Schwab, C., Ramsey, L., Wright, J., Wang, S., Gong, Q., & Roberston, P. (2016). A comprehensive radial velocity error budget for next generation Doppler spectrometers. *Ground-Based and Airborne Instrumentation for Astronomy VI*, *9908*, 99086P. https://doi.org/10.1117/12.2232761

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: a LLVM-based Python JIT compiler. *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC - LLVM '15*, 1–6. https://doi.org/10.1145/2833157.2833162

Massey, R. (2010). Charge transfer inefficiency in the Hubble Space Telescope since Servicing Mission 4. *MNRAS Letters*, *409*(1), L109–L113. https://doi.org/10.1111/j.1745-3933.2010.00959.x

Massey, R., Rhodes, J., Leauthaud, A., Capak, P., Ellis, R., Koekemoer, A., Refregier, A., Scoville, N., Taylor, J. E., Albert, J., Berge, J., Heymans, C., Johnston, D., Kneib, J., Mellier, Y., Mobasher, B., Semboloni, E., Shopbell, P., Tasca, L., & Van Waerbeke, L. (2007). COSMOS: Three-dimensional Weak Lensing and the Growth of Structure. *ApJS*, *172*(1), 239–253. https://doi.org/10.1086/516599

Massey, R., Schrabback, T., Cordes, O., Marggraf, O., Israel, H., Miller, L., Hall, D., Cropper, M., Prod'homme, T., & Niemi, S. M. (2014). An improved model of charge transfer inefficiency and correction algorithm for the Hubble Space Telescope. *MNRAS*, *439*(1), 887–907. https://doi.org/10.1093/mnras/stu012

Massey, R., Stoughton, C., Leauthaud, A., Rhodes, J., Koekemoer, A., Ellis, R., & Shaghoulian, E. (2010). Pixel-based correction for charge transfer inefficiency in the hubble space telescope advanced camera for surveys. *MNRAS*, *401*(1), 371–384. https://doi.org/10.1111/j.1365-2966.2009.15638.x

Miranda, L. J. V. (2018). PySwarms, a research-toolkit for Particle Swarm Optimization in Python. *J. Open Source Softw.*, *3*. https://doi.org/10.21105/joss.00433

Nightingale, J. W., Hayes, R. G., & Griffiths, M. (2021). 'PyAutoFit': A classy probabilistic programming language for model composition and fitting. *J. Open Source Softw.*, *6*(58), 2550. https://doi.org/10.21105/joss.02550

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Price-Whelan, A. M., Sipőcz, B. M., Günther, H. M., Lim, P. L., Crawford, S. M., Conseil, S., Shupe, D. L., Craig, M. W., Dencheva, N., Ginsburg, A., VanderPlas, J. T., Bradley, L. D., Pérez-Suárez, D., de Val-Borro, M., Paper Contributors, (Primary., Aldcroft, T. L., Cruz, K. L., Robitaille, T. P., Tollerud, E. J., … Contributors, (Astropy. (2018). The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package. *AJ*, *156*, 123. https://doi.org/10.3847/1538-3881/aabc4f

Schrabback, T., Hartlap, J., Joachimi, B., Kilbinger, M., Simon, P., Benabed, K., Bradač, M., Eifler, T., Erben, T., Fassnacht, C. D., High, F. W., Hilbert, S., Hildebrandt, H., Hoekstra, H., Kuijken, K., Marshall, P. J., Mellier, Y., Morganson, E., Schneider, P., … Velander, M. (2010). Evidence of the accelerated expansion of the Universe from weak lensing tomography with COSMOS. *A&A*, *516*(19). https://doi.org/10.1051/0004-6361/200913577

Short, A., Prod'homme, T., Weiler, M., Brown, S., & Brown, A. (2010). A fast model of radiation-induced electron trapping in CCDs for implementation in the Gaia data processing. In A. D. Holland & D. A. Dorn (Eds.), *High energy, optical, and infrared detectors for astronomy IV* (Vol. 7742, p. 774212). https://doi.org/10.1117/12.856386

Skottfelt, J., Hall, D. J., Dryer, B., Burgon, R., & Holland, A. D. (2018). C3TM: CEI CCD charge transfer model for radiation damage analysis and testing. *High Energy, Optical, and Infrared Detectors for Astronomy VIII*, *10709*. https://doi.org/10.1117/12.2309944

Speagle, J. S. (2020). dynesty: a dynamic nested sampling package for estimating Bayesian posteriors and evidences. *MNRAS*, *493*(3), 3132–3158. https://doi.org/10.1093/mnras/staa278

van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The NumPy Array2D: A structure for efficient numerical computation. *Computing in Science Engineering*, *13*(2), 22–30. https://doi.org/10.1109/MCSE.2011.37

Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., & Yu, T. (2014). Scikit-image: Image processing in python. *PeerJ*, *2*, e453.

Van Rossum, G., & Drake, F. L. (2009). *Python 3 reference manual*. CreateSpace. ISBN: 1441412697

Vehtari, A., Gelman, A., Sivula, T., Jylänki, P., Tran, D., Sahai, S., Blomstedt, P., Cunningham, J. P., Schiminovich, D., & Robert, C. P. (2020). Expectation propagation as a way of life: A framework for Bayesian inference on partitioned data. *Journal of Machine Learning Research*, *21*, 1–53. https://arxiv.org/abs/1412.4869

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Jarrod Millman, K., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … Contributors, S. 1. 0. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2