


# rushlight - Python-based Forward Modelling of Coronal Plasma Models

**Sabastian Fernandes** <sup>1\*</sup> and **Ivan Oparin** <sup>1\*</sup>

<sup>1</sup> Center For Solar-Terrestrial Research, New Jersey Institute of Technology, Newark, NJ 07102, USA   
Corresponding author \* These authors contributed equally.

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

## Software

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

Editor: [Open Journals](#) 

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished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](#))

## Summary

The rushlight Python package provides a framework for creating synthetic images of plasma structures for model-to-data comparisons with coronal events. It handles the projection and alignment of 3D simulated datasets to user-defined locations and orientations relative to the sun. The produced observables are comparable to observations made by instruments such as the Hinode X-Ray Telescope (XRT) and the Solar Dynamics Observatory Atmospheric Imaging Assembly (AIA). rushlight aims to integrate into the growing community of Python-based astrophysics software such as Astropy, SunPy and XRTpy.

## Statement of need

rushlight is a Python package which performs forward modelling of simulated 3D plasma datasets in the coronal environment. Its core functionality lies in creating synthetic observables in Soft X-Ray filter bands produced by XRT, and Ultraviolet / Extreme Ultraviolet filter bands produced by AIA.

rushlight adapts some of the core functionality of the FORWARD package, written in the Interactive Data Language (IDL) ([Gibson et al., 2016](#)). rushlight is under active development, and aims to be continually improved as to implement more of FORWARD's features.

Part of rushlight's core motivation is to make EUV / SXR forward modelling more accessible to the growing company of astrophysicists who utilize the Python language to develop and share scientific software. To this effect, rushlight has been developed as to be both compatible and scalable with release versions of other astrophysics open-source software, such as Astropy ([Astropy Collaboration et al., 2013](#)) ([Astropy Collaboration et al., 2018](#)) ([Astropy Collaboration et al., 2022](#)), SunPy ([Mumford et al., 2020](#)), and XRTpy ([Velasquez et al., 2024](#)). By creating a forward-modeling solution built upon newer and actively maintained dependencies, rushlight can be integrated into state-of-the-art solar physics research.

## Package Structure

rushlight's modules are organized as to promote the addition of new emission models and instruments to produce synthetic observables with. The package's main functionality comes from the following classes:

- `rushlight.utils.proj_imag_classified.SyntheticImage` - This module is the parent module to all other Synthetic Image classes, regardless of simulated filter type. It is responsible for translating user input into a single object containing both reference and model data. The Python module `yt` is used its ability to orient and project volumetric data from multiple simulation platforms.

- 39     ▪ `rushlight.utils.proj_imag_classified.SyntheticFilterImage` - `rushlight`
- 40     is intended to be expanded upon by developing other modules similar to
- 41     `SyntheticFilterImage`, which overloads the `SyntheticImage` class to apply the
- 42     appropriate imaging models specific to UV and SXR observations.
- 43     ▪ `rushlight.utils.dcube.Dcube` - This module serves to process user provided simulation
- 44     datasets into a `YTRegion` object. If one is not provided, it can generate a dummy uniform
- 45     grid dataset.
- 46     ▪ `rushlight.utils.rimage.ReferenceImage` - This module processes user provided
- 47     reference observation maps into `sunpy.map.Map` objects from which coordinate data is
- 48     later calculated.
- 49     ▪ `rushlight.utils.synth_tools.calc_vect` - `rushlight` accepts user specification of 3
- 50     points in 3D space located on the intended projection plane for their simulation data.
- 51     From these 3 points, it uses the simulated observer's location to calculate the vector that
- 52     is normal to this plane, and the vector that determines the rotation of the projection
- 53     relative to the normal axis. These norm and north vectors, respectively, are used in the
- 54     `yt.off_axis_projection` module to calculate projection orientation.
- 55     ▪ `rushlight.utils.emission_models.uv.UVModel` - This module is used by
- 56     `rushlight.utils.proj_imag_classified.SyntheticFilterImage` to interpolate
- 57     the temperature response function for a specified AIA channel, and then to utilize the
- 58     density and temperature data from the simulation dataset to estimate the UV intensity
- 59     of the solar plasma.
- 60     ▪ `rushlight.utils.emission_models.xrt.XRTModel` - Similar to `rushlight.utils.emission_model`
- 61     this module instead interpolates the temperature response function for a specified
- 62     combination of XRT filters to estimate the SXR intensity of the simulation dataset.

63     As referenced above, `rushlight` is dependent on a number of packages to provide critical

64     functionality.

65     The `yt` Python package ([Turk et al., 2011](#)) provides the `yt.off_axis_projection` module, which

66     is used for translating the provided 3D simulation dataset into a 2D image by efficiently

67     integrating temperature and density information along an arbitrary line of sight.

68     The `SunPy` ([Mumford et al., 2020](#)) package is the modus operandi by which the 2D observables

69     are made accessible by the package. It provides the `sunpy.map.header_helper.make_fitswcs_header`

70     module, which allows the synthetic observable's properties (such as time and coordinate

71     data) to be customized and exported as a `.fits` file along with the projected image. It also

72     provides the `sunpy.map` class, which allows easy access to the observable's properties and

73     World Coordinate System (WCS), as well as visualization of the final synthetic observable in

74     the observation plane.

75     The `Astropy` ([Astropy Collaboration et al., 2013](#)) ([Astropy Collaboration et al., 2018](#)) ([Astropy](#)

76     [Collaboration et al., 2022](#)) package is used largely for its `astropy.coordinates.SkyCoord` module,

77     which provides the flexible framework for calculating the relative positions of observables

78     in physical space, as well as transformation to and from different reference frames (eg.

79     Heliographic Stonyhurst, Heliocentric, Helioprojective). Additionally, the `astropy.units` module

80     is used extensively to dynamically determine the resulting units of various physical calculations

81     across `ruslight`'s observable alignment procedure.

82     The `CoronalLoopBuilder` ([Yu, 2023](#)) package provides a convenient way for users to define a

83     projection plane that passes through a particular point in space defined in the Heliographic

84     Stonyhurst coordinate system. `CoronalLoopBuilder` is heavily used in the Jupyter Notebook

85     example files provided with `rushlight`'s documentation, allowing the user to make dynamic

86     comparisons between their loop object, and their observable.

## Ongoing Projects

rushlight is currently being utilized to support the visualization and calculation of EUV and SXR emissions of a Magnetohydrodynamic simulation of an eruptive solar flare (Oparin et al., 2026). The synthetic observables produced by rushlight serve as the basis for comparison between the flare simulation and documented flare events, providing both a method of model validation, and a projected field of expected plasma properties that are used to further investigate the underlying magnetic field structure of the flare events.

## Acknowledgements

Test

## References

- Astropy Collaboration, Price-Whelan, A. M., Lim, P. L., Earl, N., Starkman, N., Bradley, L., Shupe, D. L., Patil, A. A., Corrales, L., Brasseur, C. E., N'otche, M., Donath, A., Tollerud, E., Morris, B. M., Ginsburg, A., Vaher, E., Weaver, B. A., Tocknell, J., Jamieson, W., ... Astropy Project Contributors. (2022). The Astropy Project: Sustaining and Growing a Community-oriented Open-source Project and the Latest Major Release (v5.0) of the Core Package. *935*(2), 167. <https://doi.org/10.3847/1538-4357/ac7c74>
- Astropy Collaboration, 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., Vand erPlas, J. T., Bradley, L. D., Pérez-Suárez, D., de Val-Borro, M., Aldcroft, T. L., Cruz, K. L., Robitaille, T. P., Tollerud, E. J., ... Astropy Contributors. (2018). The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package. *156*(3), 123. <https://doi.org/10.3847/1538-3881/aabc4f>
- 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. *558*, A33. <https://doi.org/10.1051/0004-6361/201322068>
- Gibson, S. E., Kucera, T. A., White, S. M., Dove, J. B., Fan, Y., Forland, B. C., Rachmeler, L. A., Downs, C., & Reeves, K. K. (2016). FORWARD: A toolset for multiwavelength coronal magnetometry. *Frontiers in Astronomy and Space Sciences, Volume 3 - 2016*. <https://doi.org/10.3389/fspas.2016.00008>
- Mumford, S. J., Freij, N., Christe, S., Ireland, J., Mayer, F., Hughitt, V. K., Shih, A. Y., Ryan, D. F., Liedtke, S., Pérez-Suárez, D., Chakraborty, P., K, V., Inglis, A., Pattnaik, P., Sipőcz, B., Sharma, R., Leonard, A., Stansby, D., Hewett, R., ... Murray, S. A. (2020). SunPy: A python package for solar physics. *Journal of Open Source Software*, *5*(46), 1832. <https://doi.org/10.21105/joss.01832>
- Oparin, I., Fernandes, S., Chen, B., Shen, C., Xiaocan, L., Guo, F., & Yu, S. (2026). On the nature of the candle flame shape of eruptive solar flares and sub-Alfvénic supra-arcade plasma downflows. *1*(1), 17. <https://doi.org/1>
- Turk, M. J., Smith, B. D., Oishi, J. S., Skory, S., Skillman, S. W., Abel, T., & Norman, M. L. (2011). yt: A Multi-code Analysis Toolkit for Astrophysical Simulation Data. *192*(1), 9. <https://doi.org/10.1088/0067-0049/192/1/9>
- Velasquez, J., Murphy, N. A., Reeves, K. K., Slavin, J., Weber, M., & Barnes, W. T. (2024). XRTpy: A hinode-x-ray telescope python package. *Journal of Open Source Software*, *9*(100), 6396. <https://doi.org/10.21105/joss.06396>

<sup>132</sup> Yu, S. (2023). *CoronalLoopBuilder* (Version 1.0.0). [https://github.com/sageyu123/](https://github.com/sageyu123/CoronalLoopBuilder)  
<sup>133</sup> [CoronalLoopBuilder](#)

DRAFT