




# JAX-bandflux: differentiable supernovae SALT modelling for cosmological analysis on GPUs

Samuel Alan Kossoff Leeney <sup>1,2\*</sup>

<sup>1</sup> Astrophysics Group, Cavendish Laboratory, J. J. Thomson Avenue, Cambridge CB3 0HE, UK <sup>2</sup> Kavli Institute for Cosmology, Madingley Road, Cambridge CB3 0HA, UK \* These authors contributed equally.

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

## Software

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

Editor: 

Submitted: 10 April 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Published: unpublished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

**JAX-bandflux** is a JAX (Bradbury et al., 2018) implementation of critical supernova modelling functionality for cosmological analysis. The codebase implements key components of the established library SNCosmo (Barbary et al., 2016) in a differentiable framework, offering efficient parallelisation and gradient-based optimisation capabilities through GPU acceleration. The package facilitates differentiable computation of supernova light curve measurements, supporting the inference of SALT (Kenworthy et al., 2021; Pierel et al., 2022) parameters necessary for cosmological analysis.

## Statement of need

Accurate estimation of supernova flux is essential in cosmological studies. These measurements are fundamental to the calibration of standard candles and subsequent distance determinations, which are used to answer cosmological questions. For example, the rate of expansion of the universe. Current packages such as SNCosmo (Barbary et al., 2016) are widely used for analysing supernova data. However, traditional implementations are not designed to run on GPUs and they lack differentiability. A differentiable approach enables efficient gradient propagation during parameter optimisation and supports large-scale parallel computations on modern hardware such as GPUs. This JAX implementation addresses these requirements by providing differentiable, parallelisable routines for SALT parameter extraction.

## Implementation

The package is structured into several modules and example scripts that demonstrate various aspects of the supernova modelling workflow. Two primary example scripts, `fmin_bfgs.py` and `ns.py`, illustrate optimisation via L-BFGS-B and nested sampling respectively. These scripts utilise core routines from the JAX modules, following a structure similar to SNCosmo while enabling differentiability and GPU acceleration. The central computation is contained in the file `salt3.py`, which implements the SALT3 model.

The SALT model is of the form:

$$F(p, \lambda) = x_0 [M_0(p, \lambda) + x_1 M_1(p, \lambda) + \dots] \times \exp [c \times CL(\lambda)]$$

where free parameters are:  $x_0$ ,  $x_1$ ,  $t_0$ , and  $c$ . Model surface parameters are:  $M_0(p, \lambda)$  and  $M_1(p, \lambda)$  are functions that describe the underlying flux surfaces, and  $p$  is a function of redshift and  $t - 2$ .

35 The computation of the bandflux is achieved by integrating the model flux across the applied  
36 bandpass filters. Combining multiple bands, the bandflux is defined as:

$$\text{bandflux} = \int_{\lambda_{\min}}^{\lambda_{\max}} F(\lambda) \cdot T(\lambda) \cdot \frac{\lambda}{hc} d\lambda$$

37 Here,  $T(\lambda)$  is the transmission function specific to the bandpass filter used;  $h$  and  $c$  are the  
38 Planck constant and the speed of light respectively.

39 Within `salt3.py`, the implementation computes the rest-frame model flux by combining  
40 the base spectral surface  $M_0(p, \lambda)$  with the stretch-modulated variation  $M_1(p, \lambda)$ , each  
41 scaled by their respective SALT parameters. These operations utilise JAX's vectorised array  
42 manipulations, which are JIT-compiled for efficient, parallel execution on GPUs. The resulting  
43 flux is computed in a fully differentiable manner. The computed flux is then multiplied by  
44 the instrument's transmission function  $T(\lambda)$  and by the wavelength factor  $\lambda/(hc)$ , followed  
45 by trapezoidal integration along the wavelength dimension using JAX's numerical integration  
46 capabilities. These operations are also JIT-compiled and can be parallelised across multiple  
47 data instances via `vmap`.

48 The package includes comprehensive bandpass filter handling through the `bandpasses.py`  
49 module, which provides a `Bandpass` class to represent filter transmission functions. A set of  
50 commonly used astronomical filters is pre-integrated into the system, whilst additional custom  
51 bandpasses can be registered as needed through functions such as `register_bandpass` and  
52 `load_bandpass_from_file`. The system also facilitates the creation of bandpass objects from  
53 the Spanish Virtual Observatory (SVO) filter service. For data handling, the `data.py` module  
54 offers utilities for loading and processing supernova observations from various formats, including  
55 functions to handle redshift data and prepare it for model fitting. The package currently  
56 supports both SALT3 and SALT3-NIR models through dedicated interpolation routines found  
57 in `salt3.py`.

58 This architecture allows gradient propagation through the entire analysis pipeline, enabling  
59 techniques that benefit from JAX's differentiable, parallelisable programming paradigm. The  
60 implementation maintains functional parity with `SNCosmo` whilst providing an enhanced  
61 computational efficiency and scalability for contemporary cosmological analyses.

62 Barbary, K., Barclay, T., Biswas, R., Craig, M., Feindt, U., Friesen, B., Goldstein, D., Jha,  
63 S., Rodney, S., Sofiatti, C., & others. (2016). `SNCosmo`: Python library for supernova  
64 cosmology. *Astrophysics Source Code Library*, ascl-1611.

65 Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G.,  
66 Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable  
67 transformations of Python+NumPy programs* (Version 0.3.13). [http://github.com/jax-ml/  
68 jax](http://github.com/jax-ml/jax)

69 Kenworthy, W., Jones, D., Dai, M., Kessler, R., Scolnic, D., Brout, D., Siebert, M., Pierel, J.,  
70 Dettman, K., Dimitriadis, G., & others. (2021). SALT3: An improved type ia supernova  
71 model for measuring cosmic distances. *The Astrophysical Journal*, 923(2), 265.

72 Pierel, J., Jones, D., Kenworthy, W., Dai, M., Kessler, R., Ashall, C., Do, A., Peterson, E.,  
73 Shappee, B., Siebert, M., & others. (2022). SALT3-NIR: Taking the open-source type ia  
74 supernova model to longer wavelengths for next-generation cosmological measurements.  
75 *The Astrophysical Journal*, 939(1), 11.